introducere algebrică în informatică limbaje formale

di. doc. I. CREANGĂ

(ir. C. REISCHER
profesor

universitar
dr. D. SIMOVICI
lector universitar

introducere algebrică în informatică

* *

limbaje formale

EDITURA JUNIMEA IAŞI
1974

PREFAŢĂ

Acest volum continuă seria noastră de lucrări intitulată „Introducere algebrică in informatică". El este consacrat studiului unor obiecte matematice care abstractizează proprietăţile limbajelor de programare ale calculatoarelor electronice şi ale fenomenelor lingvistice naturale.

In literatura de specialitate, care tratează probleme analoage, dată fiind dificultatea de demonstrare a unor rezultate, se recurge frecvent sau la simpla enunţare a acestora, sau la justificări aproximative. Punctul nostru de vedere a constat în pre zenlarea unor raţionamente complete în toate cazurile.

în cadrul acestei cărţi sînt trecute în revistă proprietăţile claselor de limbaje din ierarhia introdusă de Noam Chomsky, care ocupă actualmente locul central în teoria limbajelor formale. Am expus, pentru fiecare tip de limbaje, alît mijloacele finitare de generare a lor (gramatici'), cit şi dispozitivele adecvate de recunoaştere (automate).

O serie de aspecte importante, ca de pildă studiul familiilor abstracte de limbaje, utilizarea seriilor formale în studiul limbajelor, evaluarea complexităţii limbajelor, sisteme Lindenmayer, etc. vor face obiectul unui viitor volum.

Lucrarea se adresează matematicienilor, informaticienilor şi lingviştilor şi este concepută astfel, îneît lectura ei să nu impună, pe cit posibil, utilizarea altor surse. Sperăm că ea va contribui la completarea literaturii româneşti de informatică.

Autorii

— Lista de notaţii principale —

	— numărul cardinal al mulţimii M
	— mulţimea vidă
1 M /	— mulţimea părţilor lui M
0	— mulţimea părţilor nevide ale lui M
'P(M)	— funcţia caracteristică a mulţimii M
<p^*(M)	— restricţia lui 9 la I
X» <P t[1] z*	— semigrupul liber generat de 7
p. <?	— cuvinte
e	— cuvînt nul
	— cuvîntul obţinut din p prin operaţia de oglindire
	— lungimea cuvîntului p
	— mulţimea numerelor naturale
P	— mulţimea numerelor naturale
/(P)	fără zero
N	— indicele j ia valorilede la 1 la n
N^*	— laticea echivalenţelor pe S
j = 1, n	— funcţia de succesiune
Eq(S) s	— funcţia nulă
0* im	— funcţia de proiecţie
S"+1	— operator de superpoziţie
R	— operator de recursie primitivă
M	— operator de minimizare
G, Gj h	— gramatici
I_T	— alfabet neterminal
F	— alfabet terminal
	— mulţimea regulilor de generare
p => 9	— simbol iniţial
a	— cuvîntul p generează direct cuvîntul q în
$P \underset{G}{\Rightarrow} 9$	— cuvîntul p generează cuvîntul q în gramatica G

L(G) — limbajul generat de gramatica G

A «(/) — familia limbajelor de tip j, $j = 0,3$

A — mulţimea limbajelor regulate din I

I — mulţimea limbajelor finite pe alfabetul Z

S — automat

0 — alfabet de intrare

^1 | §2 — mulţime de stări

— alfabet de ieşire

— configuraţia instantanee precede imediat configuraţia

Sj |— S_2 — configuraţia instantanee S_x precede configuraţia instantanee S_2

— simbolul blanc

— simbol de marcaj

— automat pushdown

b — stare iniţială

* — limbajul recunoscut de un automat pushdown

APD — automat de tip Leticevski

^{s}o — limbaj reprezentat într-un automat de tip

L (APD) AL Leticevski

L(AL) ALM — automat liniar mărginit

L (ALM) — limbaj acceptat de un automat liniar mărginit

MT MTN — maşină Turing

MTND — maşină Turing nestaţionară

MTBI — maşină Turing nedeterministă

MTMMB — maşină Turing bilateral infinită

MT_1*MT_i *L* — maşină Turing cu mai multe benzi

(M T) — compunerea maşinilor Turing $M7$| şi MT_2

■ — limbajul acceptat de o maşină Turing

PARTEA I

ELEMENTE DE ALGEBRĂ

în această primă parte a lucrării slut expuse o serie de elemente de algebră strict necesare înţelegerii materialului prezentat. Includerea acestor paragrafe se datorează dorinţei autorilor de a face, în măsura în care acest lucru a fost posibil, ca cititorul să nu fie obligat să facă apel la alte surse.

§1. 1. NOŢIUNI DE TEORIA MULŢIMILOR

Vom expune aici cîteva elemente de teoria mulţimilor cu care ocazie vom preciza notaţiile folosite în această lucrare. Noţiunea de mulţime nu va fi definită fiind considerată aici ca o noţiune primară.

O mulţime este descrisă printr-o proprietate a clementelor sale. ■

Dacă *m* este un element din mulţimea *M* se notează acest fapt cu *m* s *M* şi se citeşte *„m* aparţine lui *M“ ;* în caz contrar vom pune *m M.*

Mulţimea M_r este *inclusă* în mulţimea M_2 dacă din *m £* $M_{\pm}$ rezultă *m* e M_2 . M_t va fi numit *parte* a lui M_2 sau *submulţime* a lui M_2 . Se scrie s M_2.

Dacă este inclus în M_2 , dar există m_2 e M_2 şi m_2 *$* M_x , atunci este *inclusă stilet* în M_2 şi se notează *cz* M_2.

Mulţimea vidă, notată cu0 este mulţimea care nu conţine nici un element. Ea este parte, prin convenţie, a oricărei mulţimi.

Dacă m s M atunci, evident, {m} c *M,* unde {m} este mulţimea al cărui unic element este *m.*

Mulţimile M_x şi M_2 sînt *egale,* $M_x = M_2$ *,* dacă M_t s M_2 şi M_2 £ M_1 .

Vom nota cu *<p (M)* mulţimea tuturor părţilor mulţimii *M.* Avem *(M)* şi *M e rp* (M); M_1 *<p (M)* dacă şi numai dacă M_1 *c M.*

Fie M_T *,* M_2 două mulţimi.

Reuniunea mulţimilor , M_2 este mulţimea

$$\text{Mj u } M_2 = [m \mid m \text{ 'G } M_x \text{ sau } m \text{ G Af}_2\}.$$

Intersecţia mulţimilor , M_2 este mulţimea

$$\text{A/\ n } M_2 = \{\text{m| m e Afj şi } m \text{ G Al}_2\}.$$

Diferenţa mulţimilor M_1, M_2 este mulţimea

$$\text{A/j \ } M_2 = \{\text{m | m e şi } m \$ M_2\} .$$

Diferenţa simetrică a mulţimilor *, M_2* este mulţimea

$$\text{A/] © } M_2 = (M_{\pm} \mid M_2) \text{ U } (M_2 \mid \text{A/,}).$$

Dacă A7j este o parte a mulţimii *M complementara mulţimii faţă de mulţimea M* sau, pe scurt, *complementara mulţimii* M_x este mulţimea

$$= \{m \mid m \text{ G } M, m \text{ AfJ}.$$

Produsul cartezian al mulţimilor , M_2 M_a este mulţimea ;

$$\text{Mj x ... X } M_n = \{(\text{fflj } m_{„}) \mid \text{m, G } M_{\text{ȷ}} \text{ , } j = 1, \text{n}\}.$$

Dacă — ... = M_n= M vom pune x ... x $M_n = M^n$.

Definiţia 1.1,1. O *relaţie n-ară* pe mulţimile Af, ,..., Af_o este o parte p a produsului cartezian M_x x ... x $M_{„}$, p s x ... x M_n. în mulţimea relaţiilor *rp* (A-/j x ... x Af_n) vom distinge relaţia vidă *0* şi relaţia totală = Mj x ... x $M_{„}$.

în cazul *= ... = $M_{„}$ = M* se obţin relaţiile n-are pe mulţimea *M.* în cazul particular n = 2 se obţin *relaţiile binare* numite pe scurt *relaţii.*

Definiţia 1.1.2. *O aplicaţie f de la mulţimea la mulţimea M_2, f:* **Afj -» M_2** este o relaţie *f £* x M_2 cu proprietatea :

$$(\text{n?!}, m_2), (m_1, m_2) \text{ G } f \text{ atunci } \text{m}_2 = m_2.$$

Vom scrie de obicei în loc de (m_L, m_2) E /", m_2 = / (m_1).

Dacă din *f* (mî) = f (m/) rezultă $m_x = mj$, *f* este o *aplicaţie injectivă* sau, pe scurt, o *injecţie* sau încă o *corespondenţă biunivocă.*

Dacă Vm_2 G M_2 există m_1 G Alj , cu *f (mf) = m_2, f* este o *aplicaţie sur- jectivă,* sau, pe scurt, o *surjecţie.*

Aplicaţia *f* este *bijectivă,* sau pe scurt o *bijecţie* dacă ea este injectivă şi surjectfvă. ’

Definiţia 1.1.3. *Proiecţia a j-a a produsului cartezian* este aplicaţia pr, : x ... x $\text{Af}_{„}$ -► *Mj* dată de

$$\text{pr}_y ((\text{Dij..m}_0)) = m_{\text{ȷ}} .$$

Definiţia 1.1.4. *O operaţie n-ară* pe mulţimea *M* este o

aplicaţie *f : M"-> M;* n este *aritatea* operaţiei *f.* ■ ■

Dacă *n* = 2 obţinem *operaţiile binare.* Rezultatul aplicării operaţiei *f* asupra Iui m_t şi m_2, *f* (m_1, m_2) se mai notează cu m_r fm_2.

Ne vom ocupa în continuare de operaţii cu relaţii.

Definiţia 1.1.5. *Produsul a două relaţii* p £ x *AL,.* T £ .17, x $A7_3$ este relaţia p-r £ A7j x .W_3 dată de :

pr = {(m! . m_3) | m_1 s AJ_i, m_3 *e* $A/_3$. 3m_2 e .17,. (mj . m_2) s p şi

(ni$_2$, m_3) E T} .

Observaţia J.l.l. Produsul de relaţii este asociativ, adică dacă pţMjX l/$_2$, a £ .17_2 x AI_3 şi - c $A7_3$ x atunci

$$P\,('T) = (pa)\ T.$$

Definiţia 1.1.6. *Relaţia diagonală* pe mulţimea *AI* este relaţia

$$= \{(m, m) | m \text{ E } A7\}.$$

Se constată imediat că. pentru orice relaţie p £ A7 x .17 avem

$$pt = ip = p.$$

Puterea a m-a a relaţiei p £ .17 x *Al* se defineşte în mod inductiv punînd p° = i, p^1 = p şi p"* = $p^{m_{-1}}$ p.

Definiţia 1.1.7. *Relaţia inversă* a relaţiei p £ .17_1 x .17, este relaţia p_{-1} c M_2 x .17_1 dată de :

$$p_{-1} = \{(m_2, m,) \mid m_2 \text{ E } A7_2,\ m_1 \text{ e } M_1,\quad . m_2) \text{ E } p\}.$$

Definiţia 1.1.8. Fie p £ .17 x .17. *Relaţia* p este :

reflexivă, dacă t £ p,
simetrică, dacă p_{-1} = p,
tranzitivă, dacă p^2 £ p,
antisimetrică, dacă p f) p'^{1} £ i.

Dacă p este reflexivă, simetrică şi tranzitivă ea este o *relaţie de echivalenţă.* Vom nota cu Eq (Al) mulţimea relaţiilor de echivalenţă pe *Al.*

Dacă p este reflexivă şi tranzitivă ea este o *relaţie de preordine.*

*" Dacă p este o relaţie de preordine antisimetrică ea este o *relaţie de ordine.*

Cuplul *(M,* p) se numeşte *mulţime parţial ordonată.*

Fie (Al, <) o mulţime parţial ordonată şi $A/_L$ £ *M.* Dacă (m', *m")* e < vom mai nota acest lucru cu *m' < m".* Mulţimea majoranţilor mulţimii A7j este mulţimea

.11} = {m | Vnqs $A7_t$, m_1 s; rn}.

Mulţimea minoranţilor mulţimii A7 este mulţimea

.17} = {m IV/HjE .17j, *m* mj.

Mulţimile A/ₓ fi AIJ şi *AIX fi AI{* conţin cel mult un element.

Să presupunem că, de exemplu mulţimea *AIₖ* fi AfJ ar conţine cel puţin două elemente fie ele *m', m".* Atunci, cum *m'* G *AIₖ* şi *m"* G Af{ avem *m'* < m". Apoi, din m" G A/ₓ şi *m'* e Alţ urmează *m"* < *m',* de unde, din antisimetria relaţiei urmează *m'* = m".

Dacă *M₁ fi Al*| / 0 *(Alₓ* fi A/J 0) unicul element al mulţimii este *cel mai mare (cel mai mic) element al mulţimii* AIX. Cel mai mare clement al unei mulţimi se mai numeşte *ultim element,* iar cel mai mic element se numeşte şi *prim element* al mulţimii.

Dacă AI* fi (A/ţ)'/ 0 atunci unicul element al acestei mulţimi este *supremumul mulţimii* A/ₓ ; îl vom nota cu sup A/ₓ.

Dacă A/{ fi (A7{)' / 0 unicul element al acestei mulţimi este *infimumul* mulţimii A/ₓ, vom nota acest element cu inf AIX.

Definiţia 1.1.9. O *partiţie* a mulţimii AI este o familie de mulţimi *{Alⱼ* |j e J] avînd următoarele proprietăţi:

1) *AIⱼ 0> Yj ᵉ J-*

2) *k* G *J, j / k* implică *Alj* fi *Alₖ* = 0,

3) U {A/, |je J} = AI.

Vom nota cu II *(Al)* mulţimea partiţiilor mulţimii *Al.*

Teorema 1.1.1. *Există o corespondenţă biunivocă intre partiţiile unei mulţimi şi relaţiile de echivalenţă pe acea mulţime.*

Demonstraţie. Fie p G Ëq *(AI).* Să considerăm familia de mulţimi TI (p) = = {p *(m)* / *m* G *Al}* unde p *(m)* = {m' | *m'* s *AI. (m, m')* G p}, Familia {p *(m)* / *m* G *Al}* formează o partiţie a mulţimii *AI.* Intr-adevăr V/n G *AI* avem p *(m)* / 0, deoarece *m* G p *(m).* Apoi, dacă (nq , *m₂)* p avem p *(mₖ) fi* p *(m₂)* =0, deoarece, in caz contrar ar exista *rn G AI* astfel ca (ni!, *m)* e p, *(m₂* , m) e p de unde, în baza proprietăţilor de simetrie, şi reflexivitate am obţine *(mᵣ, m₂)* G p.

Invers, fie. K = *{AIⱼ* |*j* G *J}* O partiţie a mulţimii *AI.* Să considerăm relaţia binară p (re) = {(/n₁, *m₂)*] 3j G *J, mₜ. in,* G *Alj},* care este' o echivalenţă aşa cum se constată imediat.

Ca atare aplicaţia A: Eq *(Al)* -»II *(AI*) dală de *k* (p) = TZ (p) este o aplicaţie surjectivă. Mai mult, ca este şi injectivă deoarece relaţia de. echivalenţă ataşată unei partiţii este, unică şi astfel *k* esle o corespondenţă biunivocă. ;

Fie *Al* o mulţime, şi p o relaţie de echivalenţă pe *AI.*

Definiţia 1.1.10. *Alulţimea cit* a mulţimii *AI* faţă de relaţia de echivalenţă p este mulţimea :

$$AI/p = \{p\ (m) \mid m \in A/\}.$$

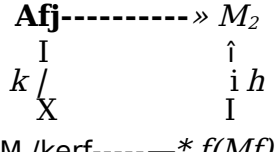
Clasa de echivalenţă a elementului *m* G AI este mulţimea p (m) şi orice element m_1 e p *(m)* se numeşlc *reprezentant* al clasei p *(m).*

Hangul relaţiei de echivalenţă p este] *AI* / p I .

Definiţia 1.1.11. *Echivalenta nucleară a aplicaţiei f* este relaţia
ker *f* £ Afj x dată de

$$\ker f = \{(fflj, m_2) \mid f(mj = f(zn_2)\}.$$

Teorema 1.1.2. *Echivalenta nucleară a aplicaţiei f este o rela(ie de echivalentă. Orice aplicaţie f:* $A/_t$ -> Af_2 *se scrie f = hgk astfel incit diagrama :*



să fie comutativă unde k este o aplicafie surjectivă, k (m) = p (m), *g este o aplicaţie bijectivă, g* (p (m)) = *f (m), iar h este aplicaţia injectivă de incluziune h (f* (m)) = *f (m),* unde p = ker *f.*

Demonstrafie. Este clar că relaţia ker *f* este reflexivă, simetrică şi tranzitivă. Apoi *h (g (k (m))) = h (g* (p (m))) = *h (f (mf) - f (m), \jm* s adică diagrama este comutativă.

Remarcăm că *g* este bine definită[1]) deoarece, dacă $p(m_1)$ = p(m) avem (m,, *m) ®* ker *f,* adică *f* (m_t) = *f* (m); *g* este surjectivă deoarece fiecărei valori *f* (m) îi corespunde o clasă de echivalenţă p *(m)* şi este injectivă deoarece din *f (mf) = f (m)* urmează (nij, *m)* s ker *f* sau p (m_L) = p (m). ■

Exemplul 1.1.1. Fie M_x s *M. Funcţia caracteristică* a mulţimii este aplicaţia y_{Ml} : A/ -> {0,1} dată de

$$X_{M,}(m) \quad \begin{matrix} 0, \text{ dacă } m \\ 1, \text{ dacă } m \in A/J \end{matrix}.$$

Remarcăm că mulţimea cit $Af/ker/_{M1}$ constă din două clase şi $A/\backslash Af_r$ Astfel rangul relaţiei ker y_{M1} este egal cu 2.

Fie *M* o mulţime şi *<Ţ> (M)* mulţimea părţilor sale.

Definiţia 1.1.12. Un *sistem de închidere* este o familie de părţi C® *<f) (M)* cu proprietăţile :

1) *\/D. D* **£ C,** Q {AI, I Af_1 e D} E **C,**

2) *M* E **C.**

Exemplul 1.1.2. Mulţimea relaţiilor de echivalenţă pe o mulţime *M* formează un sistem de închidere deoarece intersecţia unei familii oarecare de relaţii de echivalenţă este o relaţie de echivalenţă. De asemenea mulţimile relaţiilor reflexive sau tranzitive sau reflexive şi tranzitive formează sisteme de

închidere.

'> Definirea lui *g* nu depinde de reprezentantul ales al clasei p (m).

Definiţia 1.1.13. Un *operator de închidere* pe o mulţime *M* este o aplicaţie c : <Ţ> (AI) -> *(M)* cu proprietăţile :

1) AI, £ AI_2 implică e (AI,) £ c($A/_2$), VA/, , $A/_2$ & <*p* (AI).
2) AI, £ c (A/,), VAI, *e tf)* (AI),
3) c(c (A/,)) = c (A/,), VAI, e(AI).

Legătura dintre sistemele de închidere şi operatorii de închidere este precizată în

Teorema 1.1.3. *Orice sistem de închidere* **C** *defineşte un operator de închidere prin*

(1.1.1) e_c (L) = n {A/, | Al, s **C,** *L* £ AI,}.

Invers, orice operator de închidere **e** *pe M defineşte un sistem de închidere* $\mathbf{C_c}$ = {AI, | AI, e *M,* **c** (A/,) = A/,}. *Aplicaţia f : SI -» OI, unde SI este mulţimea sistemelor de închidere ale mulţimii M. OI mulţimea operatorilor de închidere şi f* **(C) =** $\mathbf{c_c}$ *este o bijecţie.*

Demonstraţie. în baza egalităţii (1.1.1) operatorul e_c satisface primele două condiţii din definiţia 1.1.13. Apoi e_c (A/,) = A/, dacă şi numai dacă A/, e **C.** într-adevăr dacă *e* **C** cum M_1 *£* A/, avem c_c (A/,) £ A/, şi proprietatea 2) implică $\mathbf{c_c}$ (A/,) = AI,, iar dacă $\mathbf{c_c}$ (A/,) = A/, avem A/, e **C** deoarece **C** este sistem de închidere şi c_c (A/,) e **C.**

De aceea, cum c_c (L) e C, |/*L* e <*p* (A/) obţinem e_c (c_c (L)) = e_c (L), adică c_c este un operator de închidere.

Invers, fie c un operator de închidere şi fie **D £** $\mathbf{C_c}$ o familie de mulţimi. Să arătăm că

c (n {AI, I AI, G **D}) =** n {AI, I A/, e **D).**

Va fi suficient să verificăm că **c (** (1 {A/, | A/, e D}) £ n {AI, | A/, s **D}.** Avem **c** (n {A/, | AI, e **D}) £ c** (A/,) = A/,, e **D.** De aici obţinem incluziunea dorită.

în fine c (A/) = A/ deoarece *M* £ c (AI) £ *M.*

Aplicaţia *f: SZ -» OI* este evident surjectivă în baza celor de mai sus. Să arătăm că ea este injectivă adică e_{Ci} = c_{CJ} implică C, = C_2 . Va fi suficient să arătăm că C_{Cc} = C.

într-adevăr C_{Cc}= {AI, | AI, £ AI, e_c (A/,) = AI,}. în baza unei observaţii făcute în cursul demonstraţiei avem $\mathbf{C_{Cc}}$ £C. Dacă AI, e **C** avem $\mathbf{c_c}$ (AI,) = AI, adică AI, e $\mathbf{C_{Cc}}$**.** ■

Exemplul 1.1.3. Fie $\mathbf{C_{fIT}}$ sistemul de închidere al relaţiilor reflexive şi tranzitive pe mulţimea AI şi e operatorul de închidere corespunzător. Atunci

$$\mathbf{e\ (a) = U}$$
1-o

unde a £ .V x *AL* Avem 01 a-" £ a° = şi (0 c?) (0 o?) £ ti *a.'* deci

0) a'.'= C_{nr} şi a £ 0 a'. De aceea e (a) £ 0 Fie acum (ni'. *m")* <= x>. J =o Există {m* | Ael,j—1} £ *AI* astfel incit {(ni', nij), ($n^{}$, m_2),... ($m_{/_1}$, ni")} £ £a£e(a). Cum e (a) este o relaţie Iranzitivă obţinem (m', zh")ec(a) şi dacă

a^7 £ e (a),' V./ ^{e} A de unde 0 o? £ e (a). relaţia

Definiţia 1.1.14. *închiderea reflexivă* şi *tranzitivă* a relaţiei *y.* esle

Noţiunea de graf dc care ne vom ocupa in continuare repr permite o zentare „geometrică" pentru relaţii. e-

Definiţia I.l.lo. Un *graf orientat* este o pereche ® = (V, F) esle unde o mulţime, iar F o aplicaţie, T: V->rp(V). Mulţimea V se *mulţi numeşte, mea vîrf urilor grafului* ® sau *mulţimea nodurilor- grafului* ® iar dacă v_2 e F *(vf)* perechea ordonată *(v_{lt},* a_2) este *ărc f-*

Dacă] V J < oo graful ® este *finii;* în caz contrar ® esLc un *graf infinit.* care uneşte punctele v_t şi v_2 îndreptată de la

Grafurile se reprezintă ataşînd fiecărui vîrf un punct din plan şi fiecărui arc . v_2) o săgeată

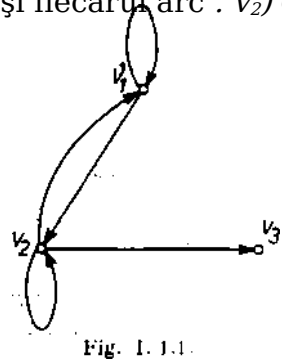


Fig. I. 1.1

Graful (ţa2.a3}. AJ

Definiţia 1.1.17. *Mulţimea (arcelor* grafului ® A (V, T) *U* £ V x V dată de *U* (a, a_g) .în arcul (a,, a_g) vîrful v Aste *extremitatea iniţială* a arcului (a_x, a_2).

Exemplul 1.1.4. Fie ® = ({PJ , v_2, a_3}, T) unde F este dat de T (aj = {a_t, a_2}, r(a_2) = {a_1, a_2, a_3}. r *(v_3) = 0.* Graful ® este reprezentat în figura 1.1.1.

Definiţia 1.1.16. Un *subgraf al grafului* (V, T) esle un graf (V_x, Tj) cu Vj£ V şi Tj (a) = T (a) D V_r

Un *graf parjial* al grafului (V, T) este un graf dc forma (V, A) cu A (a) £ F (a), |/v *e* V.

Un *subgraf parţial* al grafului *(V,* T) esle un graf (V_1, AJ cu Vj £ V şi A_x (a) £ F (a) Q .

Exemplul 1.1.5. Fie ® = ({PJ , a_2, a_3}, T) graful din exemplul 1.1.4. Graful ({p', a_2}, Tj) cu Tj *(0^)* = Tj (a_2) = {a_x, a_2} este un

graf parţial al un subgraf

Subgraful grafului ®.

Fie graful ({PJ, a_2, a_3}, A) cu A (aj = {PJ}, A (a2) = {a_3}, A (a_3) = 0 este un subgraf al grafului (V, T). Aici v_t *extremitatea finală* a arcului (a_2, a_3). Mulţimea ($v_{„_l}$, a_B) cu a' = v_L şi a„ = a". Un *cir-*

cuil este un drum pentru care Pj = v_n . Numărul n-1 este *lungimea drumului.* L[T]n circuit de lungime 1 se numeşte *buclă.*

Definiţia 1.1.19. Un *graf conex* este un graf ® = (V, T) pentru care oricare ar fi vii turile *v', v"* G V cu *v' / v"* există un drum de la *v'* la *v"* sau de la *v"* la *v'.* Graful este *tare conex* dacă oricare ar fi vîrfurile *v', v"* G V cu *v'* p" există un drum de la *v'* la *u".*

Definiţia 1.1.20. *Componenta conexă* a vîrfului *v* este mulţimea de vîrfuri C_v = {p} U {p' | există un drum de la *v'* la *v* sau de la *v'* la p}.

Observaţia 1.1.2. Să considerăm relaţia p £ V x V dată de

Se constată imediat că p este o relaţie de echivalenţă şi că V/p = = |C_r | P G V}. Astfel componentele conexe ale vîrfurilor formează o partiţie a lui V.

Pentru un graf conex avem | V'/p| = 1.

Definiţia 1.1.21. Un *arbore* este un graf conex ® = (V, T), fără circuite, avind | V | 2 şi care satisface la următoarele două condiţii :

i) există un virf P_0 care nu este extremitate finală a vreunui arc ; P_0 se numeşte *rădăcina arborelui ;*

ii) orice vîrf p, P G V\{P_0} este extremitate finală a unui singur arc. Nodurile p pentru care T (P) *= 0* se numesc *noduri finale.*

Exemplul 1.1.6. Fie graful ® = ({P_0 , P_4 , P_2 , P_3 , P_4 , p_5}, T) cu T (P_0) = = {p,. pj. **r** (PJ) = {P_2 . P_3}. **r** (P_2) = r (P_3) = 0, **r** (P_4) = {P_s}. **r** (P_s) = 0, reprezentat in fig. 1.1.2.

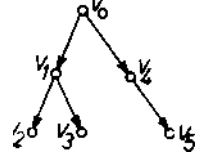


Fig. I. 1.2.

Graful ® este un arbore avind rădăcina P_0 şi vîrfurile finale P_2 , P_3 şi P_5 .

Definiţia 1.1.22. Vîrful P_2 este *descendent* direct al vîrfului P, în arborele ® = (V, T) dacă (Pj , P_2) [5] Tu {(,} ; cu alte cuvinte P_2 este un descendent direct al lui v_L dacă există arcul (P_x , P_2) în graf, sau P_2 = Pj ; în primul caz P_2 este un *descendent propriu,* iar în al doilea caz un *descendent impropriu* al vîrfului *Pj .*

Vîrful p" este *descendent* al vîrfului *v'* în arborele S = (V, T) dacă există un şir de vîrfuri Pj . P_2 P„ astfel incit $P_{;+1}$ este descendent direct al vîr
fului *i>j . j* = 1, *n* — 1, P' = Pj şi p„ = P".

Dacă (P_0 . Pj) (Pj , P_2),..., (P_{n-1} . P_n) este drumul care uneşte vîrful P = P_n cu rădăcina P_0 a arborelui atunci *n* este *înălţimea vîrfului v.*

Observaţia 1.1.3. Orice graf ® = (V, T) defineşte o relaţie pg> £ V x V

prin $(v_1, v_2) \in \rho_{(G}$ dacă şi numai dacă $v_2 \in \Gamma(v_1)$.

Invers, pentru orice relaţie p s V x V există un graf ®$_p$ = (V, r_p) care defineşte această relaţie : T_p este dată de

$$r_P\ (y) = \{"i\ I\ (".\ "i)\ ^e\ ?\}\bullet$$

Definiţia 1.1.23. Un *graf marcat* este un 4-uplu ® = (V, T, *I,* p) unde (V, T) este un graf, *I* este *mulţimea simbolurilor de marcare,* iar

$$R : \{(p, , P_2) \mid P, e\ F\ (PJ\} \text{ -> } I.$$

este *aplicaţia de marcare.*

Exemplul 1.1.7. Fie graful marcat ® = (V, T, /, p) unde (V, T) este graful dinfig. 1.1.1, / = {O, 1}, p ($(p_{\pm}$, pj) = O, p ($(p_L$, P_2)) = 1, p ($(P_2$, Pj))=O, p ($(P_2$, P_2)) = O, p ($(P_2$, P_3)) = 1. Acest graf marcat este reprezentat în fig. 1.1.3 indicîndu-se pe fiecare arc simbolul de *marcaj* corespunzător.

Noţiunea de graf este generalizată în

Definiţia 1.1.24. Un *hipergraf* este o pereche *j) = (X. E)* unde X este o mulţime numită mulţimea *virfurilor hipergrafului,* iar *E* este o familie de părţi ale lui X. *E = [E_f* I je J}cu proprietăţile :

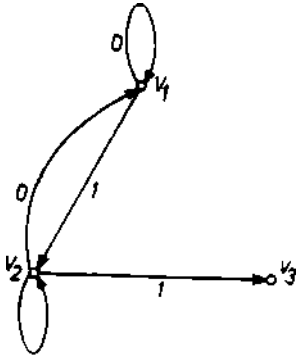


Fig. 1. 1.8.

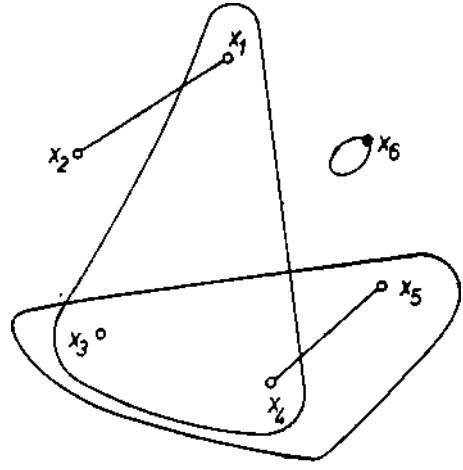


Fig. 1. 1.4.

1) *Ej* / 0, VJ ^{s} *J*

2) u {Kj |*j* ^{e} •/} = -V numite *arce ale hipergrafului.*

Dacă | *X*I < oo hipergraful este finit ; în caz contrar este un hipergraf infinit.

Observaţia 1.1.4. Oricărui graf ® = (V, F) i se poaLe ataşa un hipergraf

J5 = (V, /?}), arcele hipergrafului fiind mulţimile date de

i) {p}, dacă $C_i = \{p\}$

ii) {PJ , P2}, dacă v_2 s r(i'i).

Exemplul I.l.S.Vie hipergraful 55 = ({x_4 , x_2, x_3 , x_4 , x_s, x_6}, {Tîj , E_2 , L_3, 7:4, E_s}) unde E_4 =■ {x_4, x_3 , x_4}, E_2 — {x_3 , x_4, x_5}, E_3 - - {x_4 , .r_5},

= {x_6}, 7i_5 = {x_4, x_2}. Acest hipcrgraf este reprezentat în figura **1.1.4.**

Definiţia 1.1.25. Un *drum de lungime n înlr-un hipergraf* 55 = **(X,** *E)* este un şir {Xj , E_4, x_2 , E_2 ,..., E_n , x_{o+1}} astfel încît :

i) Xj,.... x„ sînt vîrfuri distincte ;

ii) E_4,..., E_n s *E* sînt arce distincte ;

iii) x_t. x_{t+1} *e E*| pentru k = 1, n.

Pentru n > 1 şi x_{B+1} = x_4 obţinem circuite de lungime *n.*

Observaţia 1.1.5. Relaţia p c X x X dată de (x_4, x_2) p dacă şi numai dacă există un drum în, hipergraful 55 = (X, *E)* care pleacă din vîrful x, la vîrful x_2 este o relaţie de echivalenţă. Clasele de echivalenţă corespunzătoare se numesc *componentele conexe ale hipergrafului.* Hipergraful este conex dacă | X/p |=1.

Definiţia 1.1.26. *Virfurile interioare* ale arcului E_1 sînt vîrfurile din mulţimea 7^\u {7:* | *k / j}* ; celelalte vîrfuri ale arcului *Ej* sînt *virfurile conectate* ale arcului.

Definiţia 1.1.27. Un *hiperarbore* este un hipergraf conex **55 = (X,** *E),* fără circuite, avînd | X | 2 şi care satisface la următoarele condiţii :

i) există un vîrf x_0 care nu este vîrf conectat al vreunui arc ; x_0 se numeşte *rădăcina hiperarborelui;*

ii) orice vîrf conectai al unui ăn arc. are cel mult un vîrf conectat pentru exact

Vîrfurile neconectalc diferite de x_0 sînt *virfurile terminale* ale arborelui.

Exemplul 1.1.9. Hipcrarborele x_4 , x_5 , x_6 ; x_7}, {E_1 , E_2 , E_3 , E_4}) unde E_4 ≡ {x_0 , x_1 si E_4}= E_1 = {x_1, x_3 , x_4}, E_3 = {x_3, x_5} şi E_4 = {x_4 , x_6, x-} are ca rădăcină vîrful x_0, ca noduri conectate x_4, x_3, x_4, iar ca noduri terminale x_2, x_5. x_6, x_7 . Acest hiperarbore este reprezentat în figura 1.1.5.

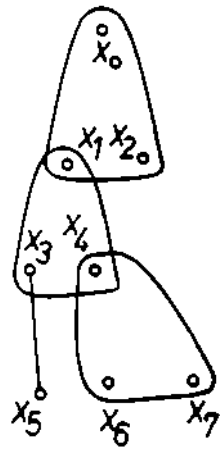


Fig. I. 1.5

Definiţia 1.1.28. Vîrful x" este *descendent al vir- fului x'* în hipergraful 55 = (X, *E)* dacă există un drum de la x' la *x" ;* dacă acest drum are lungimea unu sau dacăx" = x' at unci x" este un *descendent imediat* al lui x'. Un descendent *x"* al lui x' cu *x"* / x' este un *descendent propriu,* în caz contrar un *descendent impropriu.*

Ultima parte a acestui paragraf este consacrată prezentării noţiunii de număr cardinal.

Fie *T* o mulţime , pe care o vom numi în

y = {(A^ , Al_2) | Alj , M_2 G (p (T), 3f: M_k -» M_t, f aplicaţie bijectivă}.

Teorema 1.1.4. *Relaţia* y *este o relaţie de echivalenţă.*

Demonstraţie. Deoarece aplicaţia 1_M: M -»M este o bijecţie urinează *(M, M)* y,. W ^{s} <P (T) adică y este o relaţie reflexivă.

Dacă (Afj, M_2) ey există bijecţia $f:M_1$-^M_2. Atunci există bijecţia f^{-1} : Al_2 —> M_l şi deci *{M2, Mf)* e y, adică y este o relaţie simetrică.

Fie (M_k, M_2), (M_2, M_3) E y. Există bijecţiile *f*: A/j -> M_2 şi *g : M_2* -» Al_3. Deoarece aplicaţia este şi ea o bijecţie avem *(M*j, $A7_3$) sy

deci y este tranzitivă.

Definiţia 1.1.29. Un *număr cardinal* este o clasă de echivalenţă din mulţimea <7>(T),/y. *Numărul cardinal al mulţimii M* este numărul cardinal căruia îi aparţine *M.* Vom nota acest număr cu (*M |.*

Definiţia 1.1.30. O *mulţime infinit numărabilă* este o mulţime .1/ pentru care (Al, N) s y, unde *N* este mulţimea numerelor naturale. O *mulţime finită* este o mulţime Al pentru care există *n N* astfel ca (Al. {l,...,n}) <s y. Notăm în acest caz | Al] = n şi convenim că |*0* | =0.

O *mulţime numărabilă* este o mulţime finită sau infinit numărabilă.

Observaţia 1.1.6. Orice parte a unei mulţimi numărabile este o mulţime numărabilă.

Exemplul 1.1.10. Mulţimea numerelor pare *P* este infinit numărabilă deoarece aplicaţia *f:N*->-P* dată de *f* (n) = 2n este o bijecţie.

Exemplul 1.1.11. Mulţimea perechilor ordonate de numere naturale Al = {(m, n) i *m, n N*}* este infinit numărabilă. Fie $A/_t$ = {(m, n) | *m + n =*

= k 4- 1} ; *M* = (J M_k. Mulţimea M_k conţine *k* elemente deoarece *m* poate s-i

lua valori de la 1 la *k.* Astfel aplicaţia p : Al -> *N* dată de p ((m, n)) =

(m + *n* — 2) (m + *n* — 1) . .. .

= ---------- ---------- 4- *m* este o aplicaţie bijectivă. Ea indica

locul prerechii *(m, n)* în şirul

(1,1) (1,2), (2,1), (1,3), (2,2), (3,1), ...

în care apar succesiv elementele mulţimilor M_k şi în care perechile fiecărei mulţimi M_k sînt plasate în ordinea crescătoare a primei lor componente.

Să notăm m 4- n — 1 = A. Vom arăta că există un număr A, unic, cu proprietatea :

2 2

Din inegalităţile (1.1.1) obţinem :

$\frac{\sqrt{1+8h}-1}{2} < A \leq \frac{\sqrt{1+8h}+1}{2}$ con-

l[un]8'[mea] 1 [e]l [nl] şi cum intervalul $*' -\!\!-^{are}$ celo

ţine decît un singur număr natural.

Astfel p ((m. n)) = — — + *m.* Oricare ar fi *h* G *N** în baza arătate mai sus se poate determina *A* în mod unic astfel ca (3 ((m, n)) deoarece în acest caz

$$\frac{A\,(A-1)}{2\ 2} < h < \frac{A\,(A+1)}{2}$$

Astfel zn = *h*------ ------- şi n = A — *m* + 1, de unde rezultă şi unici-

tatea perechii (zn, n) cu proprietatea p ((zn, n)) = *h.* Conchidem că aplicaţia 3 este bijectivă şi aslfel aplicaţia p^{-1} : N* -»■ *Al* realizează „numărarea" elementelor mulţimii *AI.*

Rezultă de aici că dacă *Al* este o mulţime infinit numărabilă atunci mulţimea $Al^2 = AI$ x *AI* este infinit numărabilă.

Exemplul 1.1.12. Mulţimea numerelor raţionale *Q* este infinit numărabilă deoarece ea poate fi pusă în corespondenţă biunivocă cu mulţimea perechilor ordonate de numere naturale. Astfel fie *r* un număr raţional. Există în mod unic perechea de numere naturale (p, *q),* prime între ele astfel incit r *= p/q.* Aplicaţia y : Q-» N_x cu c *N x N, A*| fiind'format din acele perechi de numere naturale, prime între ele, dată de y (r) = y *(p/q)* = (p, *q)* este o bijecţie.

Nu orice mulţime este numărabilă, aşa cum rezultă din

Exemplul 1.1.13. Mulţimea numerelor reale din intervalul [0, 1] nu este numărabilă. într-adevăr, să presupunem că ea ar fi numărabilă şi fie {r, IjejV} = [0, 1] o numerotare a elementelor mulţimii [0, 1].

Orice *r* G [0, 1] se poate scrie *r* = 0, a_t, a_2,..., a_n ... cu a_n G {0, 1,...,9}. Să notăm *r (m)* = a_m , *m* G *N** şi să considerăm numărul real

$$r' = 0,\ p_{,},\ p_2 \ldots p_n \ldots$$

cu p_j / (J). Evident *r'* G [0, 1] şi oricare ar fi *j, r' r*$_j$. Astfel şirul *[rj* |j G A^7} nu epuizează mulţimea [0, 1] aşa cum era presupus iniţial.

Exemplul 1.1.14. Mulţimea de funcţii *F = {[/ f: N* -» {0, 1}} nu este numărată avînd acelaşi număr cardinal cu mulţimea [0, 1]. Fie *x = 0, x*$_t$*, x*$_2$ *... ... x*$_n$ *...* un număr din intervalul [0, 1] scris

în baza 2. Avem x_t G {0, 1}. Aplicaţia a : [0, 1] -» *F* cu a *(x) = f*$_x$, unde f_x (zi) = *x* (n) este bijectivă deoarece dacă f_x (n) = f_v (n), V>· $^{1\,s}$ *N* avem *x = y* şi pentru orice funcţie *f* G *F* există *x* G [0, 1] *cu*f*$_x$ *= f.*

Deoarece *F* c *SJ / f*: N -» A'} urmează că nici mulţimea *{f| f: N* -» A^7} nu este numărabilă.

Teorema 1.1.5. *Fie [AI, / j* G A^7} *o mulţime numărabilă de mulţimi numă- rabile. Atunci Al =* u {A/, |j G A^7} *este mulţime numărabilă.*

Demonstraţie. Deoarece mulţimea M_j este numărabilă putem scrie M_j = = {cj, r_2 r* ,...} şi astfel M_j este pusă în corespondenţă biunivocă cu mulţimea de perechi {(j, 1), *(j,* 2),..., (j, n),...}. în consecinţă mulţimea *M* este în corespondenţă biunivocă cu mulţimea perechilor ordonale de numere naturale adică ea este numărabilă [v. exemplul I.l.ll].|

Teorema 1.1.6. *Mulţimea părţilor finite a unei mulţimi numărabile este o mulţime numărabilă.*

Demonstraţie. Fie *M* o mulţime numărabilă. Să arătăm mai înlîi că mulţimea M^k este numărabilă, procedînd prin inducţie după *k.* Pentru *k* 2 afirmaţia este adevărată (v. şi exemplul 1.1.11).

Să presupunem că M^k este o mulţime numărabilă. Alunei cum Af^{t+1} este în corespondenţă biunivocă cu mulţimea M^k x *M* prin aplicaţia a : : 3/*$^{+1}$ M^k x *M* dată de

$$a ((nij,..., m_{t+1})) = ((m_{\pm},.... m_k), m_{t+l})$$

rezultă că M^{k+1} este numărabilă.

în baza teoremei 1.1.5 mulţimea M_l = u *{M*^{k} / *k* JV} este o mulţime numărabilă. Mulţimea părţilor finite ale mulţimii *M* este în corespondenţă biunivocă cu mulţimea cît / p unde p = {((m, mj, (mj,..., m^)) | ^{1} {m_x= {m, m*}}. ■

§ I. 2. LATICI

Definiţia 1.2.1. O *latice* este o mulţime *M* împreună cu două operaţii binare pe *M,* reuniunea şi intersecţia notate respectiv cu u şi fi care se bucură de următoarele proprietăţi :

1) m_1 u m_a = m_2 u $m_{\pm}$, m_L n m_2 = m_2 n m_t , \/m_2 , m_2 *e M* (co muta tivitate),

2) nij u *(m₂* u m₃) = (™i U *m₂)* u *m₃, mₜ* CI (m₂ fi m₃) = (m₁ n n *m₂)* n *m₃, Vmₓ, m₂, m₃ e M* (asociativitate),

3) mj u (m₂ 0 *m₃)* = m₁, *mₜ* n *(m₂* U m₁)=m₁, V™i. m₂ *e M* (absorbţie).

Vom nota laticea astfel definită cu *(M,* u , 0).

Observaţia 1.2.1. Remarcăm că în sistemul de axiome din definiţia precedentă putem schimba semnele u şi n între ele. Ca atare, acest lucru se poate face în orice afirmaţie dedusă din acest sistem de axiome, proprietate cunoscută sub numele de *principiul dualităţii pentru latici.*

Exemplul 1.2.1. Mulţimea *<P(M)* a părţilor unei mulţimi *M* împreună cu reuniunea şi intersecţia formează o latice.

Mulţimea numerelor naturale cu operaţiile

$$^{n}\text{i u } n_2 = [\text{Hi}, n_2],$$
$$\text{Hj A } n_2 = (\text{nj}, n_2),$$

unde [n^, n₂] şi , n₂) înseamnă respectiv cel mai mic multiplu comun şi cel mai mare divizor comun al numerelor şi n₂ formează o latice.

Observaţia 1.2.2. Avem

m U m = m şi m A m *= m,* ym e *M* (proprietale de idempotenţă).

Intr-adevăr, folosind de două ori proprietatea de absorbţie găsim mai întii *m* u (m A m) = *m,* de unde *m* 0 [m u (m 0 m)J =m n *m* sau *m = m* A m. Similar se arată că m = *m* u *m.*

Teorema 1.2.1. *Fie (M,* u . A) *o latice. Următoarele două afirmaţii sint echivalente :*

$$\text{i) } m_2 = m_Y \text{ u } m_2 ,$$
$$\textit{ii)} \text{ nij} = /\text{Hj A } m_2.$$

Demonstraţie. Dacă *m₂ = mᵧ* u *m₂* din proprietatea de absorbţie găsim *m± = mᵧ* A (mj u m₂) = m₁ n m₂, adică i) => ii). Implicaţia ii) => i) se arată analog.

Teorema 1.2.2. *Relaţia*

$$= \{(\text{ni!}, m_2) \mid m_1 = m_{\ell} \text{ A } m_2\},$$

este o relaţie de ordine parţială. Mai mult mₜ u *m₂* = sup {nij , m₂} *şi mₜ* A *m₂ =* = inf {fflj, m₂}.

Demonstraţie. Belaţia < este reflexivă (deoarece *m = m* A *m),* antisi- inetrică (căci *mₖ - mₜ* A *m₂* şi *m₂* = m₂ A m₁ implică mₓ = m₂) şi tranzitivă (pentru că A *m₂* şi *m₂ = m₂* A *m₃* atrage *mₜ* = mᵣ A *(m₂* A *m₃) =*
= (m₁ A m₂) n m₃ = fflj n m₃).

Avem *mₓ* u *m₂* = {m₁, m₂}* A ({//^ , m₂}')(. într-adevăr *m₂ <*

m_t u $^{m}{}_2$ deoarece $m_1 = m_t$ A (fflj u *rn₂)* şi similar $m_2 < m_1$ u m_2. Apoi, dacă m, < m şi m_2 < m obţinem $m = m_2$ u m şi $m = m_2$ u m în baza teoremei 1.2.1, de unde $m = (m_t$ u m) U $(m_2$ U m) = $(m_l$ u $m_2)$ u m sau u m_2<m, adică m_2 u m_2 *e* ({m_1, $m_2)'y$. Asemănător obţinem m_x A m_2= inf {m_1,m_2}. ■

Teorema 1.2.3. *Fie (M, <) o mulţime parţial ordonată. Să presupunem că oricare ar fi m_{ℓ}, m_2* e *M există* sup {m_1, m,} şi inf (THJ , m_2}. *Punind*

$$m_L \; u \; rn_{\;2} = \sup \{fflj, m_2\},$$
$$n\hat{} \; A \; m_2 \;\; \inf \{n1_2, zn_2\},$$

mulţimea M cu operaţiile u *şi* A *este o latice.*

Demonstraţie. Vom verifica axiomele laticelor din definiţia 1.2.1.

Evident sup {nij, m_2} = sup {m_2, mj, de unde m_i u $m_2 = m_2$ u m_1 . Apoi inf {fflj, m_2} = inf {m_2, m,} de unde A $m_2 = m_2$ A m_t .

Să notăm

m' = sup {m_1, sup (m_2, m_3}} şi m'' = sup {sup {m,, m_2}, m_3}.

Avem m' şi m' > sup {m_2, m_3} sau m' > *m, , m_2, m_3* . De aici m' >
> sup {nij, m_2} şi deci m' m". Similar obţinem m'' m' de unde m'' = m' şi rezultă asociativitatea reuniunii.

Demonstrarea asociativităţii intersecţiei este asemănătoare.

în sfîrşit, fie m = sup {m_t, inf {m_2, mj}. Atunci m > m,, Cum m_l m_1 şi m_l inf {m_2, mj avem > sup {rzij , inf {m_2. mj} = *m.* Astfel m = , adică obţinem nij u (m_2 n *mf)* = m_1 . La fel se arată că m_{ℓ} n (m, u mj = = şi obţinem proprietatea de absorbţie. ■

Consecinfa 1.2.1. Fie *M* o latice faţă de operaţiile u şi P. şi relaţia de ordine parţială definită în teorema 1.2.2. Operaţiile obţinute in teorema 1.2.3 coincid cu operaţiile iniţiale.

Astfel o latice poate fi definită fie cu ajutorul operaţiilor de reuniune şi intersecţie, fie cu ajutorul unei relaţii de ordine parţială pentru care există sup {m_1, m_2} şi inf {mj, m_2} pentru orice pereche de elemente *in_L, m_2 e* Af.

Definiţia 1.2.2. O *latice completă* este o mulţime parţial ordonată (Af, <) pentru care există sup M_t şi inf M_x , oricare ar fi A7$_1$ g *M.* Vom nota sup **.11, =** = U {m | m s A/J şi inf = 0 {wj | m_1 *e MJ.*

Evident, orice latice completă este o latice, dar nu orice latice este completă.

Exemplul 1.2.2, Mulţimea *<J)ţM)* a părţilor unei mulţimi *M* este o latice completă faţă de relaţia de incluziune dintre mulţimi.

în schimb mulţimea (0, 1) este o latice faţă de relaţia de

ordine obişnuită dintre numerele reale, dar nu este o latice completă deoarece sup (0, 1) — = 1 £ (0,1) şi inf (0, 1) = 0 £ (0, 1).

Teorema 1.2.4. *Mulţimea parţial ordonată (M, este o latice completă dacă una din următoarele două afirmaţii este adevărată :*

1) *există* inf M_t, *yXf £ M şi mulţimea M conţine un cel mai mare element ;*

2) *există* sup *£ M şi mulţimea M conţine un cel mai mic ele ment.*

Demonstraţie. Vom arăta că, în primul caz, există totdeauna sup M_y, yM_t s *M.* Deoarece mulţimea *M* conţine un cel mai mare element, mulţimea. M_2 = {*m* / *m* s *M, m* m_t, ym_ℓ e A/J este nevidă. Fie *m'* - inf M_2. Să arătăm că *m'* = sup $M_{\pm}$.

Dacă m_t s M_1 atunci oricare ar fi *m* s M_2 avem *m rn*$_1$, adică m_Y e .Vj . Ca atare inf M_2 = *m' m*$_t$, V^i ^{s} •

Apoi, fie m" > m_1, \/m_t s Afj ; atunci *m"* e M_2 şi astfel *m"* inf M_2. Conchidem că inf Af_2 = sup .

Similar se arată că şi condiţia a doua este suficientă pentru ca mulţimea parţial ordonată (Al, să fie o latice completă.

Consecinţa 1.2.2. Un sistem de închidere **C** c *<Ţ>(M)* esle o latice completă fiind închis la intersecţii oarecare şi conţinînd mulţimea *M.*

Exemplul 1.2.3. Mulţimea Eq *(M)* este o latice completă în baza consecinţei 1.2.2. Atragem atenţia asupra faptului că reuniunea în această latice nu coincide cu reuniunea relaţiilor considerate ca părţi ale mulţimii *M* x *M.* în general, dacă Ej.este o mulţime de echivalenţe sup E, c (j {o:|a e Ej}.

Definiţia 1.2.3. O *lalice mărginită* este o latice *M* care posedă un cel mai mic şi un cel mai mare element ; le vom nota respectiv cu 0 şi 1. Avem. Vm S *M ;*

$$0 < m\ 1,$$

$$0\ u\ m = 1\ A\ m = m,$$

$$0nm=0;lu^{\wedge} = l.$$

Observaţia 1.2.3. O latice completă este mărginită luînd 1 ca intersecţia familiei vide de mulţimi, iar 0 ca reuniunea aceleiaşi familii.

Teorema 1.2.5. *(teorema de punct fix a lui Knăsler-Tarski). Fie (M, o latice completă şi f : M -> M o aplicaţie izotonă (\/*m_i *.* m_2 **® A/,** m_t m_2 *avem f (mf) f* (m_2)). *Există rn* s Af *cu f (m) = m.*

Demonstraţie. Să considerăm mulţimea = {mi | m_1 Af , m_t < *f* (m,)}. Avem A/j / *0* deoarece 0 s . Fie *m* = sup . Dacă m_t s AZ,

atunci $m_{\pm} < m$, de unde $m_r\ f\ (m_t) < f\ (m)$. Urmează de aici că m = sup A/j f (m) adică m s .

Prin urmare $f\ (rri) < f\ (f$ (m)), ceea ce arată că $f\ (m)$ s . în baza definiţiei lui m avem atunci $f\ (rri)\ m.$ de unde $f\ (m) = m.$ Astfel m este un punct fix al aplicaţiei considerate. ■

Definiţia 1.2.4. Fie *(M,* u , A) o latice mărginită şi m e A/. Un *element complementar* (pe scurt un *complement)* al elementului m este elementul m' cu $m\ v\ m' = 1$ şi m **A** $rri = 0$.

Observaţia 1.2.4. Nu orice element dinlr-o latice mărginită are un complement. Astfel, în laticea mărginită ({0, *m.* 1}, u , A) elementul m nu are complement.

Complementul unui element, dacă există, nu este în mod necesar unic. Astfel în laticea ({0, , m_2, m_a, 1} , u , A) în care operaţiile sînt date în
tabloul de mai jos :

u	0	THj	77^2	m_3	1	A	0	mi	m_2	m_3	1
0	0	mi	m_2	m_3	1	0	0	0	0	0	0
ml	mi	"h	m_2	1	1	mi	0	mj	nii	0	mi
m_2	m_2	m_2	7?	1	1	77?2	0	mi	m_2	0	
m_3	m_3	1	1	m_3	1	m_3	0	0	0	m_3	m_3
1	1	1	1	1	1	1	0		m_2	m_3	1

elementul m_3 are două complemente : pe m_t şi pe m_2.

Remarcăm că elementele 0 şi 1 au fiecare complement unic, respectiv pe 1 şi 0.

Definiţia 1.2.5. O *latice complementară* este o latice mărginită în care orice element m are un complement unic m^e.

Definiţia 1.2.6. O *latice distributivă* (.W, u, A) este o latice în care *\/m₁, m₂. m₃ e M* avem :

(1.2.1) (mj u m_2) n m_3 = *(m_L* n *m_3)* u *(m_2* n m_3).

Observaţia 1.2.5. într-o latice distributivă are loc şi egalitatea

(nij n m_2) u m_3 = *(m_t* u m_3) n (m, u m_3),

care se obţine din (1.2.1) aplicînd principiul dualităţii.

Teorema 1.2.6. *într-o latice distributivă mărginită complementul unui element, dacă există, este unic.*

Demonstraţie. Să presupunem că

$$m \text{ n } m' = m \text{ n } m'' = 0,$$

$$m \text{ u } m' =\bullet m \text{ u } m'' = 1.$$

Atunci $m' = m'$ n 1 = m' A *(m* u m") = *(m'* n *m)* u (m' n m") = = m' n m'' adică $m'm''$. La fel se arată că m'' m' de unde m' = m".H

Definiţia 1.2.7. O *algebră Boole* este o latice distributivă şi complementară, în baza teoremei 1.2.6 într-o algebră Boole complementul unui element m este unic şi va fi notat cu m^e. Se poate concepe o algebră Boole ca un 4-uplu (.W, u, n,^{e}) în care. ^{e} este operaţia unară de complimentariere.

Exemplul 1.2.4. Mulţimea u , n /) formează o algebră Boole
in care .Wţ = Af\M_1.

Exemplul 1.2.5. Algebra Boole cu două elemente este B_2 = ({0, 1}, u , n ,^{c}) în care operaţiile sînt date mai jos :

u	^{0}1	n	^{0}1	'	1 0 1
0	0 1	0	00	1	1 0
1	1 1	1	01		

Ea poate servi la construcţia corpului finit cu două elemente, notat cu *GF* (2). Mai precis, considerind operaţiile de adunare *x ® y* = (x (1 y®) U U (x* n y) şi de înmulţire *xy = x* n y pe mulţimea {0, 1} obţinem un corp.

Teorema 1.2.7. *într-o algebră Boole au loc relaţiile lui De*

(1.2.
2)

Morgan

$$(n?!\ u\ m_2 y = n\ m^e{}_t.$$

$$(njj\ n\ rn_2 y = rrtt\ u\ m' \qquad {}_{t-}$$

Demonstraţie. Vom dovedi că (mj u m_2) u (m_2 0 mj) = 1 şi (m_Y u m_2) n n (ruj n nij) = 0 de unde, ţinînd seama de unicitatea complementului într-o Iatice distributivă vom obţine formula (1.2.2). Avem

$$(m, u\ m_2)\ u\ (mţ\ 0\ mj) = ((m_t\ u\ m_2)\ u\ mţ)\ 0\ ((m_l\ u\ m_2)\ u\ m'_2) =$$

$$= 10\ 1=1,$$

iar

$$(m_1\ u\ m_2)\ n\ (mţ\ n\ m^c{}_2) = (m_x\ n\ (mj\ n\ m£))\ (J\ (m_2\ n\ (mţ\ n\ m£)) =$$

$$= 0\ U\ o = 0.$$

Similar se procedează pentru (1.2.3). ■

Noţiunea de Iatice se poate generaliza. Astfel are loc

Definiţia 1.2.8. O *semilatice* este o mulţime *M* prevăzută cu o operaţie care satisface următoarele axiome :

1) $V^m i,\ m_2{}^e \qquad {}^m i * m_2 = m_2 * m_1$,

2) $V^m i > {}^m 2 > {}^m 3\ ^e\ Af, \quad {}^m i * ({}^m 2 * {}^m a) = ({}^m i\ ' {}^m 2) * {}^m 3\ '$

3) Vm e Af, m ' m = *m.*

• - w

Observaţia 1.2,6. Orice Iatice *(M,* u . n) este semilatice faţă de fiecare din cele două operaţii.

§ I. 3. SEMIGRUPURI

Definiţia 1.3.1. Un *semigrup* este un cuplu (Al, *) în care *M* este o mulţime. nevidă, iar este o operaţie binară asociativă ; adică *\/m*r, *m*2, *m*3 *e M* are loc egalitatea :

$$fflj * (m_2 * m_3) = (mj * m_2) * m_3 .$$

Un *semigrup cu unitate (monoid)* este un triplet *(M,* *, e) în care (Af, •) este un semigrup, iar *e e Al* are proprietatea :

$$e * m = m \text{ » } e = m, \backslash/m \text{ e Af},$$

şi este numit *unitatea semigrupului.*

Exemplul 1.3.1. Mulţimea numerelor pare formează semigrup faţă de operaţia de înmulţire.

Tripletul *(N,* +, 0) este un monoid.

Exemplul 1.3.2. Orice semilatice *(M,* u) este un semigrup numit *semigrup semilaticial.* Dacă această semilatice are prim element atunci *(M,* u . 0) este *monoid semilaticial.*

Definiţia 1.3.2. O *congruenţă la dreapta (stingă)* pe semigrupul *(M, »)* este o echivalenţă p s .17 x *AI* cu proprietatea : *(m₁ , m₂)* ˢ p => *(m₁* m, m₂ * m)* e p (=> *(m * , m* ♦ m₂) s p), ym e *M.*

O *congruenţă* este o echivalenţă care este simultan congruenţă la dreapta şi la stingă.

Observaţia 1.3.1. Intersecţia unei familii oarecare de congruenţe (congruenţe la stingă sau la dreapta) este o congruenţă (congruenţă la stingă sau la dreapta).

Astfel mulţimea congruenţelor (congruenţelor la stingă, la dreapta) formează un sistem de închidere.

Fie *1* o mulţime cel mult numărabilă numită *alfabet.* O *literă (un simbol)* este un element oarecare *i* E 7.

Definiţia 1.3.3. Un cuvînt peste alfabetul **7** este o aplicaţie *p:* {1,..., *n} ->I.* Aplicaţia *p* se notează de obicei prin *p = iₓ... i„ ,* unde *iⱼ = p (j), j =* 1, *n.*

Lungimea cuvîntului p este *l (p) =* / (i\ ... *i„) = n,* adică numărul sim- bolilor din *I* care apar în cuvîntul *p.*

în mulţimea cuvintelor se introduce operaţia de concatenare (juxtapunere). Fie *p* şi *q* două cuvinte peste acelaşi alfabet 7, *p = iₜ... iₘ , q =* ij ... i' .

Definiţia 1.3.4. *Cuvîntul obţinut prin concatenare* din cuvintele p şi q este cuvîntul *pq = iᵧ... iₘ i[... i; .*

Avem *l (pq) = l(p) + l (q).*

Observaţia 1.3.2. Mulţimea tuturor cuvintelor peste un alfabet 7, dotată cu operaţia de concatenare este un semigrup numit *semigrupul liber* generat de *I,* notat cu 7*.

în semigrupul liber *I** există elementul neutru la operaţia de concatenare : *cuvîntul nul e,* cu *l (e) =* 0. Avem *ep = pe = p, \/p* s 7*.

Denumirea de „semigrup liber" este justificată prin :

Teorema 1.3.1. *O aplicaţie* cp : *I -> S, unde S este un semigrup oarecare se extinde în mod unic la un homomorfism* 9ᵣ: 7* -► S.

Demonstraţie. Fie pe 7*. Vom defini qpj *(p) =* cp₁ *(i₂ ...* 9 (ij) 9 (<₂)...
... 9 *(i„).* Se constată imediat că tpₓ astfel obţinut este un

homomorfism de semigrupuri care coincide pe generatorii semigrupului liber *I** cu aplicaţia 9.

Să presupunem că ar exista un homomorfism 9_2 : 7* -» S astfel incit ?2 ţ 7 = 91 [■ 7.

Mulţimea *E* definită prin *E* = {p | 9] (p) = 9_2 (p)} este un subsemigrup al lui 7*.

Să observăm mai iutii că *E* ± 0 deoarece 7 c 7?.

Fie p, q E *E.* Atunci ?1 *(pq)* = ?1 (p) ?1 *(q)* = <p_2 (p) ?2 *(q)* = <p_2 *(pq),* deci *pq E E.*

Cum *E* este un semigrup care conţine mulţimea de generatori ai semigrupului 7*, urmează *E = I*.* ■

Definiţia 1.3.5. Un *limbaj* pe alfabetul / este o parte a semigrupului liber *I*.*

Mulţimea limbajelor pe alfabetul *I,* este o algebră Boole faţă de
operaţiile obişnuite din teoria mulţimilor, de reuniune, intersecţie şi coin- pliinentariere.

Vom utiliza *produsul* dintre două limbaje. Astfel, produsul limbajelor şi E_2 este limbajul

$$\text{Zij } E_2 = \{p \mid p = \text{Pj } p_2 \text{ , Pj } ^{e} E_L \text{ . } p_2 \text{ e } l\text{^}\}\text{-}$$

Produsul de limbaje este asociativ, dar nu comutativ. Limbajul {e} joacă rolul elementului neutru la produs, {e} *E* = *E* {e} = *E.*

Limbajul vid, 0 este element nul faţă de aceeaşi operaţie. 0/< = F0 = 0. *închiderea prin produs* a limbajului *E* este limbajul

$$A^* = \bigcup_{k-0}$$

unde E° = |e}, iar $E^k = E\ E^{k\sim^{l}}$. \/*k* e A'*.

§ I. 4. SPATII METRICE ŞI SPAŢII LINEARE

Fie *M o* mulţime oarecare.

Definiţia 1.4.1. O *metrică* pe mulţimea .V este o aplicaţie d : A/ x *M* -» *R* care are următoarele proprietăţi:

1) d (ni] . m_2) > 0, \/m_i , m_2 E Af şi d (04 , m_2) = 0 dacă şi numai dacă

$$m_x = m_2;$$

2) d (mj , m_2) = d (m_2, mj. \/m_i . m_2 s *M,*

3) d (m_1 , m_3) *4* d (mj, m_2) 4 d (m_2, m,), V4 , ni_2, m_3 e

Perechea (3f, d) se numeşte *spaţiu metric.*

Definiţia 1.4.2. Un *spaţiu liniar real* sau pe scurt, un *spaţiu liniar* este o mulţime *M* prevăzută cu două operaţii :

1) o operaţie internă, adunarea faţă de care *M* formează un grup abe- lian, adică sînt satisfăcute axiomele :

la) nij 4 (m_2 4 m_3) = (mj 4 m_2) 4 $^{In}3$- • $^{III}2$ > $^{m}3$ ^{e} (asociativitate),

lb) 3o e A/ astfel incit *o* 4 *m* = *m* 4 *o* — *m.* \/*m e* Af (existenţa elementului nul),

lc) Pentru *m* e *M,* 3m' E *M* cu *m* + ni' = m' 4 ni = o. Voin nota *w'* = -m (existenţa elementului opus).

ld) m, r m_2 = m_2 4- m_1, V$^{/?,}$i < ^{,n}*t* *= *M* (comutalivitatc).

2) o operaţie externă,' înmulţirea cu elementele din *R* faţă de care sini satisfăcute axiomele :

2a) a (m, -i- m_2) = am, 4 a/n_2, Va e R m, . m_2 E *M,*

2b) (a 4 |3) *m* = am 4 |3m, Va, {3 E R, m E Af,

2c) a (Şm) = (a|3) *m,* Va, (3 E *R. m* E Af.

2d) lm = m, Vm E Af,

Elementele lui *M* se numesc vectori, elementele lui *R* scalari, iar cele două operaţii se numesc respectiv adunare de vectori şi înmulţirea cu scalari.

Exemplul 1.4.1. Mulţimea numerelor complexe *C* formează un spaţiu liniar faţă de adunarea numerelor complexe şi înmulţirea cu numere reale.

Definiţia 1.4.3. O *normă* pe spaţiul liniar *M* este o aplicaţie || . ||: *M* ->■ *R* care are următoarele proprietăţi :

1) // *m* || > 0, Vm E *M* şi || *m* // = 0 dacă şi numai dacă *m* = o.

2) II m, 4 m_2 || < || m_l || 4 j| m_2 j| y/n, < ^{in}i ^{s} -^{v}-

3) ||am || = | a j |1 *m* ||, ya E *R, m* E Af.

Norma vectorului m este numărul ||m||. Cuplul *(M,* || . ||) se numeşte *spaţiu liniar normat.*

Exemplul 1.4.2. Aplicaţia | . | : *C R* dată de | *a*^*-bi* | = /a^2 4 b^2 este o normă pe *C.*

Teorema 1.4.1. *Eie* (Af, || . ||) *un spaţiu liniar normat. Aplicaţia* d : *M* x Af -+ *R dală prin* d *(m_L, m_2) =* || m, - m_2 || *este o*

metrică pe M.

Demonstraţie. Avem d (m, ¸ m_2) > 0 deoarece Hm, — m_2 l| > 0. Apoi, (1 (m, . m_2) = 0 dacă şi numai dacă || m, — m_2|| = 0, adică dacă şi numai dacă m, — m_2 = *o* sau m, = m_2.

Simetria metricii se obţine imediat : d (m, . m_2) = || m, — m_2 1| = || (-1). . (m_2 - m,) ||=|| m_2 - m_t || = d (m_2, m,).

în sfîrşit d (m,, m_3) = || mj — m_3|| = || m_1 —m_2 4 m_2 —m_3|| 4 'I m, —
- $m_2$1| + I' m_2- $m_3$1| = d (m_t . m_2) 4 d (m_2, m_3). ■

Teorema precedentă justifică

Definiţia 1.4.4. *Metrica indusă de norma* || . ||pe spaţiul liniarnormat (Af, || . ||) este aplicaţia d : Af x Af-» *R* dală de.d (ni!, m_2) = || m_l — m_2||.

Exemplul 1.4.3. Metrica indusă de norma considerată în exemplul 1.4.2 este dată **de**

$$d(\ll 4\ bi,\ c - di) = J((_a\ _\ _c)2\ 4\ \text{ţ}b - d)^2.$$

l'ic acum (Af, d) un spaţiu metric.

Definiţia 1.4.5. *Şirul* {m_B | n G A'*} c *M* este *convergent* şi *are limita m* G Af dacă, oricare ar fi e G R_t e > 0 există n_0 G Ă' astfel incit n n_0 implică d *(m_n , m)* < e.

Definiţia 1.4.6. Şirul {$m_„$ | n G A^7*} £ ,V este *şir Cauchy* dacă oricare ar fi *z G R,* £> 0 există n_0 G A^T astfel incit *n* n_0 implică d *(m_n* , m_{B+jl})< e, oricare ar fi *p* e *N.*

Observaţia 1.4.1. Un şir convergent este şir Cauchy. înlr-adevăr fie {m_B | *n* G A'*} un şir convergent avind limita *rn şi z' = z/2.* Există n_0 astfel incit dacă *n* n_0 avem d (m_B , *m) < z'.* d (m_{B+jl}, m) < z^1. *Vp e N.* Folosind proprietăţile metricii găsim d ($m_„$, m_{n+p}) *d (m,* , m) + d (ni, $ni_{B+j)}$) < 2e' = *z.*

Reciproca acestei afirmaţii nu este adevărată aşa cum rezultă din :

Exemplul 1.4.4. Fie spaţiul metric ((0, 1], d) unde d *(x, y) = x — y.* Şirul

- |n G A* l este un şir Cauchv, deoarece dacă *n* >-avem---------

$$\frac{p}{\frac{1}{\varepsilon}\left(\frac{1}{\varepsilon}+p\right)} = \frac{p}{p+\frac{1}{\varepsilon}} \varepsilon <$$

n

n | ' e n n + p

este şir convergent in cadrul spaţiului metric specificat. Dacă am considera Spaţiul metric ([0, 1], d) şirul ar deveni convergent deoarece limita sa este 0 G [0, 1J.

Apare astfel justificată :

Definiţia 1.4.7. Un *spaţiu metric complet* este un spaţiu metric în care orice şir Cauchy este un şir convergent.

Fie (A/, d) un spaţiu metric şi .1 : Af -» .V o aplicaţie ; vom nota cu *Am* elementul A (ni).

Teorema 1.4.2. *(teorema de punct fix a lui 13anach). Dacă (M,* d) *este un spaţiu metric complet şi dacă \/m_l , m_2* G .W *avem* d (Am_t, .lm_2) < *a* d *(m_l, rn_2), unde a* < 1 *atunci există in M un element unic m astfel incit Am = m ; m este punctul fix al operatorului* A.

Demonstraţie. Fie m_0 G Af ; formăm şirul

$$UR = Am_g\ ,\ m_2 = A/HJ\, m_{B+1} = Ani_„\ ,\ldots$$

Avem d (m_2, mj = d (Am_v An^) < *a* (d (m,, *n^).* în general d (m_{B+1} , $m_„$) = d ($Am_„$, *$Am_{„_1}$) a^n* d (nij . m_0).

Mai departe obţinem

$$m_B) < d\,(m_{B+j>}, m_{B+1_1}) \text{ -i- } d\,lm,_{+p}\hat{}_i \,.\, m_{„+p_2}) - \ldots -$$

$$\text{-r- } d\,(m_{n+i}\,,\, m_{„}) < (a^n \text{ a}^{"+1} \mid \ldots \qquad d\,(m,\,,\, ni_0) =$$

$$= a^n \frac{1 - a^p}{1 - a} d\,(m_1\,.\,m_0) < \frac{a^n}{1 - a} d\,(nij\,.$$

C.um $a < 1$ avem lim d ($m_{„+il}$, m,) = 0, adică şirul [m, | n s A"} este şir n-»ao Cauchy. Spaţiul (A/, d) fiind complet şirul este convergent şi fie m limita Iui. Vom arăta că $Am = m$.

Avem d *(Am. m)* d *(Am,* $m_{„}$)- d *($m_{„}$,* m)= d (Am. A/i^Jt d *(m_n, m)* < *a* d *(m. m_{n_l})* 4 d *(m_n . m).*

Deoarece limm„ = *m* oricare ar fi e e K_+ există n_0 e A' astfel că dacă *n* -> x

n_0 avem *d(m„,n)*< - e şi d *(m, m. _,)*<-e. Alunei d *(Am. m)* <- - ■ - 2 2 2 2 = s ceea ce implică d (Am. *m)* - 0 adică *Am = m.*

Punctul fix *m* al aplicaţiei *A* este unic. înlr-adevăr să presupunem că ar mai exista un punct fix *m'* al aplicaţiei .4. Atunci d *(m, m') =* d *(Am, Am')* 4 î? *a* d *(m, m'),* adică (1 — *a)* d *(m, m')* < 0, ceea ce nu se poate întîmpla decît dacă d *(m. m') =* 0 adică *m = m'.* ■

O aplicaţie .4 : .1/ — .1/ cu proprietatea din enunţul teoremei se numeşte *aplicaţie contractantă* sau *operator contractant* pe spaţiul metric *M.*

Obseri>a(ia 1.4.2. Teorema lui Banach îşi menţine valabilitatea dacă operatorul A este contractant numai pe mulţimile .4* (.V). *li* A'*.

§ I. 5. SPAJIUL METRIC AL LIMBAJELOR

Fie *I* un alfabet finit. Vom organiza mulţimea *ep (I*)* ca un spaţiu metric. In acest scop vom introduce pe *<Ţ> (I*)* o structură de spaţiu liniar normat şi vom considera apoi metrica indusă de normă.

Teorema 1.5-1. *Mulţimea <P(I*) este un spaţiu liniar peste corpul GF* (2) = = {0. 1}, *faţă de operaţia internă :*

$$E, E_2 = (E, \text{ n } E_t) \text{ u } (E' \text{ n } E_2) = (E_A \text{ U } E_2) \mid (E, \text{ o } E_2)\hat{}$$

si operaţia externă, de înmulţire cu scalari : 0 . E = E . O = 0 şi 1 *. E = = E .* 1 = *E.*

Demonstraţie. Se verifică uşor că *((p* (/*), -j-) este un grup aditiv abe- Iian în care elementul nul este 0, iar opusul lui *E* este tot *E* şi că :

$$a\ (E_J\ 4\text{-}\ E_2) = aE_L\ 4\text{-}\ a.E_2\ .$$

$$(a.\ 4\ p)\ E_Y = xE\text{^}\ \text{-}4\ p/ij\ .$$

Va. p *GF* (2). *E, ,E₂^ep (/*)•■*

⁺) Ej4-este diferenţa simetrică a mulţimilor şi 7.₂ ■

Teorema 1.5.2. *Aplicaţia || . ||* : *<p (/*) -» R dală de || E* |j = 2-^(E> . *unde l (E)* = inin *{l* (p) | p E *E] esle o normă pe <Ţ> (/*). \'om lua 1(0)* = oc.

Demonstraţie. Vom verifica faptul că aplicaţia ||.|| salisface proprietăţile unei norme.

1) Avem |j *E*|| > 0. *\/E* E *<p* (7*) şi || *E* || = 0 dacă şi numai dacă *E* = 0.

2) **jj** *v.E* || = I a | l| *E* ||. V» ᵉ *GF* (2) şi *F* *= T⁵ U*)- Aici I ᵃ I ’ᵃ valorile reale O sau 1 după cum a esle unul din elementele O sau 1 ale corpului *GF* (2).

3) || *E,* 7I₂|| < // *E,* || +||E₂||.

Pentru a demonstra această relaţie să observăm că, dacă *E £ E'* atunci // *E* || sg || *E'*|| şi |! *E_t* u *E..W* = max {|| 7-, ||. || *E₂*1|}. deoarece *l (E_t* u 7?,) = min *l(E₂)}.*

Atunci || 77, *E₂* || F ,i 7-.’, u *E₂* || = max (||//, // *E₂*1|) sg || *E_t* || -f- || *E₂*1|. ■

Consecinţa 1.5.1. Spaţiul normat ('jD (7*), 4 . . . || . ||) este un spaţiu metric avînd metrica:

$$p(7\text{-}, i/\text{-}:_2) = 117'.',— E_2||=||E, F/<_2 1).$$

Vom caracteriza in continuare şirurile Cauchy în acest spaţiu.

Teorema 1.5.3. *Şirul* |*E_n* | *n* E A⁷| *este şir Cauchy dacă şi numai dacă* 7* *putem avea pentru orice cuvint p* E *următoarele două si numai una din tuaţii :*

1) 3n (p)E A'*astfel incit pentru Vn n (p) avem p* E 7'.',, .

2) 3n (p)s A’*astfel incit pentru \/n n (p) avem p £ E_n .*

Demonstraţie. Să presupunem că şirul *{E_n* | *n* E A⁷*} este şir Cauchy. Atunci Ve > O, 3)'E E X astfel ca || *E_{n+m}* - *E_n*|| < s de îndată ce /? > *n. m* £ A’*.

Fie p E 7* şi s sg Atunci p nu poate aparţine numai unuia din limbajele *E_{n + m}* şi *E_n* pent ru *n* > *n_z* deoarece în acest caz am avea *l (E_{n+m}* — *E_m) = i (E„_{+m}* + 7-;„) F / (p), de unde // *E_{n+m}* - *E_n* ||

> 2 '<p> F e. Aşadar luînd /i *(p) = /;£* I 1. p se poale afla numai într-una din cefe două situaţii de mai sus.

Putem să arătăm acum că dacă şirul {/■.'„ | *n E* A'*} are proprietatea din enunţ atunci el este şir Cauchy.

Fie s > 0. Considerăm limbajul E_t = *{p* | p E 7*, / (p) <c - loji, e}. care esle evident un limbaj finit.

Pentru orice p E E_e există *n* (p) astfel îneît să avem sau p E E_n sau p $ E_n pentru *n > n* (p).

Fie n_z = max {n (p) | p E E_e*}.* Atunci, pentru *n* > $n_{£}$. p $ E_{n+m} F *E„. \/ms.* A^T, de unde *l (Em+„ + En) = l (Em^n- E„)* > - lo<>$_2$ s. adică || 7-.'$_{n+m}$- - F_n || < e, pentru *n* > n, . ■

Observaţia 1.5.1. Din teorema precedentă rezultă că nu orice şir din spaţiul considerat este şir Cauchy.

Teorema 1.5.4. *Spaţiul metric (Ţ>* (7*) *esle complet.*

Demonstraţie. Fie *{En* | n E A'} un şir Cauchy oarecare. Vom arăta existenţa unui limbaj *E* astfel incit liin p *(En , E) = (),* Astfel, să luăm *E* =
= U OE$_e$, limbaj alcătuit din acele cuvinte *p* care se află în toate
limba- 1 A *m*
jele şirului *{En / n* e N*} începînd de la un anumit rang, adică *E* = lim inf *{Ek / k* e A/*}.

Fie e > 0 şi limbajul finit *E = {p| pel| l (p) sg* — $\log_2$ e}. Ca şi în demonstraţia teoremei precedente se arată că există n_e astfel încît pentru *n* > n_t avem *p e En* sau *p £ E„ ,* oricare ar fi *p* s *Et .* Mai mult, *p En,* pentru n > n_t pentru orice *p* e *Et* Q *E.* Apoi, dacă *p^Et* avemp^E—£„ , n > n_g şi deci *l (E — En) >* — $\log_2$ e sau || *E — En* || < e pentru *n* > n_e. ■

Consecinţa 1.5.2. Orice şir ascendent de limbaje {E_{fl} | *n e* N*}, *(En* c E_{n+1} , yn e N*) este convergent şi lim *En* = (J *En .*
n *n*

Un şir ascendent este şir Cauchy, deoarece dacă *p e I** atunci sau există n_p s N* astfel ca *p* e *En?* (şi deci *p e Em , \/m, m* n_p) sau *p* (J *En.*

în baza teoremei 1.5.4, limita şirului ascendent {E_n | *n e N*}* este *E* = Ob OD 00

= u n^ = u *E„.*

m—1 n—1 *m-l*

Fie *1' = I\J* {y}, *y £ /, Te rpţl'*) şi E e fpţl*).*

Definiţia 1.5.1. *Operatorul* T : -> *indus* de limbajul *T e* este aplicaţia care asociază fiecărui limbaj *E e(Ţ>* (/*) limbajul T (E) obţinut din *E* prin înlocuirea în cuvintele lui *T* a literei *y* prin toate cuvintele limbajului *E.*

Exemplul 1.5.1. Fie / = {i_3 , i_2}> *E* = {ij, *i1 i2, i2* ij, *T* = {ij... *i± y i2... i2*

n *n*

n e N}. Atunci T *(E)* = {ij... i_x i_2... i_2 , ii... ii i_2... i_2. <i ••• ii '2 L '2 ••• '2 n + 1 © » + l o + l *n n / n e N}.*

Observaţia 1.5.2. Operatorul T : -> ^(P) este izoton, adică E_1 £ E_2 =>
T (E>) £ T (E_2).

Fie *T e <p(T*).* Limbajul *T* se poate scrie sub forma :

(1.5.1) $T = T_t$,

unde T, = T n {y}*. *T,* = T Q = T\ {T_vJ TJ.

Observaţia 1.5.3. Pentru orice E s *fpţl*)* avem:

(1-5.2) l|T,(E)U T„(E)|| < ||E||,

deoarece prin înlocuirea simbolului *y* în cuvintele din 7j|J T_{iy} prin cuvinte din E se obţin cuvinte de lungime cel puţin egală cu cea a cuvintelor iniţiale, dacă e E. Dacă *e e E* atunci || E || = 1 şi în acest caz are loc de asemenea inegalitatea (1.5.2).

Definiţia 1.5.2. Un operator nesingular este un operator T pentru care T_y *=0.*

Observaţia 1.5.4. Dacă operatorul T este nesingular atunci || (E) (J

Mai mult, putem afirma că $\| \mathbf{T}_{ty}(E) \| \leqslant \frac{1}{2} \| E \|$.

Teorema 1.5.5. *Un operator nesingular* T *este contractant. Demonstraţie.* Este suficient să

(1.5.3) T (/•:,) 4 T(E2)|| + E2||, VE, . E2G <?)(/.).

Să observăm că dacă T este nesingular atunci :

(1.5.4) jT (/•;,) + T (E2)|| = T^EJ r T,(Ex) + Tf,(E2) + T,(E2) = = ^(Ei) 4- T(I/(E2)

deoarece l'^Ej) = T_t (E2) şi deci $T_{(}$ *(Ek) + T(* *(E2) - 0.*

Fie *r* G (Ex) (- T((, *(E2).* El se obţine în urma înlocuirii intr-un cuvînl oarecare $p_o y p_t$ ... $y p_k$ « T_{tv} , p_j G /*, *j* = 0, *k* a simbolului *y* prin cuvintele , g_2,..., g_k respectiv, unde *cp , j* = 1, *k* aparţin toate sau lui E_k sau lui E2 , deci *r* - p_0 <*p* p_Y ... *ip* p_k. Nu putem avea *(p* G E\ Q E2, *j* = 1, *k* deoarece in acest caz *r* G- T(„ *(Ef)* Q T(s/ (E2) şi deci *r* T{J, *(Ef)* + Tfi, (E2). Prin urmare există un indice *J,* 1 *j* 4 *k* astfel încît *Qj* G E_k, dar *cp* E2 sau invers, deci q_j G Ex + E2.

Deoarece avem în mod necesar max{/(p/),j = 0, *k}* 1 urmează că *l (r) l (cp),* ceea ce înseamnă că pentru orice cuvînt din *T(u (Ek*) 4- *T(u (E2)* există un cuvînt de lungime mai mică în E_1 4- E2 de unde urmează (1.5.3). ■

Observaţia 1.5.5. Pentru un operator T singular (cu T_y / 0) nu mai avem în mod necesar max *{l (pf) \j =0, k}* 1 şi în locul inegalităţii (1.5.3) avem

(1.5.5) || I (E,) -|- T (E2) || < || Ex + E2 ||.

sau încă

(1-5.6) II **T** (E\) - T (E2) || 4 |i E\ - E2||.

Teorema 1.5.6. *Orice operator* **T** *deforma introdusă in definiţia* 1.5.1 *este continuu.*

Demonstraţie. Rezultă imediat din observaţia precedentă. ■

Consecinţa 1.5.3. Fie E = lim E_n şi E' = lim E'_n şi **T** (En) c **T** *(E'n)* pentru *n* > nx. Atunci **T** (E) £ **T** (li').

într-adevăr, din continuitatea operatorului **T** avem lim **T** (E„) - **T** (E) şi lim **T** *(E'„)* = **T** (E'). Dacă *r* G **T** (E) atunci există n_2 G N* astfel ca *n*
r G **T** (En), pentru n > n2. Luînd n_3 = max (/ix, n_2) avem *r* G **T** *(E'n)* pentru n > n_3 , de unde *r* G **T** (E').

Ne vom ocupa în continuare de studiul ecuaţiei :

(1.5.7) $E = \mathbf{T}(E)$

Existenţa unei soluţii pentru ecuaţia (1.5.7) rezultă din teorema 1.2.5 de punct fix a lui Knăster-Tarski, deoarece este o latice completă iar
T este un operator izoton.

Structura de spaţiu metric şi teorema 1.4.2 de punct fix a lui Banacli ne vor permite să aducem precizări suplimentare.

Teorema 1.5.7. *Limbajul E =* lim $\mathbf{T}^4$ (K), *unde K* <p(/*) *şi* $\mathbf{T}^4$ *(K) =* **= T(T**$^{4-1}$ (K)) **(T**^{1} (K) = *T(K))este"o soluţie a'ecuaţiei* (1.5.7).

Demonstraţie. Dacă operatorul T este nesingular atunci el este contractant şi afirmaţia este valabilă în baza teoremei de punct fix a lui Banach.

Fie deci **T** un operator singular. Dacă operatorul **T** este contractant pe mulţimile $\mathbf{T}^4$ *(K),* $\mathbf{T}^{4+1}$ Ti > 1 atunci teorema este adevărată aşa cum
rezultă din observaţia 1.4.2. Pentru a verifica proprietatea de contracţie pe mulţimile $\mathbf{T}^4$ *(K).* $\mathbf{T}^{4+1}$ **(/<),...,** *h* 1 va fi suficient să arătăm că

(1.5.8) $\| \mathbf{T}^{m+1}\ (K) - \mathbf{T}^{4+}\ (K)\| < I\| \mathbf{T}^{m}\ (K) - \mathbf{T}^{4}\ (K)\|.$

pentru *m, n* **e** A*, cu *m, n h.*

Fie *r* e T^{m+1} (k) -|- T^{4+1} *(K).* Atunci *r = p0 q{ pk q2... qk pk* şi există cu- vîntul *p0 ypk y ... ypk* e *Ty* u *Tiy* din care obţinem cuvîntul *r* prin înlocuirea lui *y* prin cuvintele *qk qk* E J-» (/<) u T^4 *(K).* Să observăm că există un *j,* 1 *< j < k* astfel încît pentru cuvîntul *qj* să avem sau r/, s T^m (K) şi *qj* T^4 *(K)* sau T"' (7<) şi <= T^4 (A) deoarece în caz contrar *r*
^T^{m+I} (K) + $\mathbf{T}^{4+1}$ *(K).* Dacă pentru Vm, *n >h* şi pentru orice cuvînt *r* există totdeauna un *qj* cu *l* (r) > *l (qj)* atunci inegalitatea (1.5.8) este satisfăcută şi operatorul **T** este contractant pe mulţimile $\mathbf{T}^4$ *(K),* $\mathbf{T}^{4+1}$ (K),... Vom arăta că şi în cazul cînd inegalitatea (1.5.8) nu-i satisfăcută, ecuaţia (1.5.7) are soluţie. în această situaţie, are loc egalitatea *l* (r) = *l (q^* de unde urmează */■ = qj* ceea ce implică *p0 = pk = ... = pk = e* şi *e* E **T"'‘** *(K)* (j **T"** *(K).* Dar dacă e e **T'** *(K),* cu *l* > 1 atunci *e* E *T* sau *e* E *K* deoarece operatorul **T** este singular. De aici *e E* **T** *(K),* şi în general *e* E **T"**! *(K), \/m* E *N** şi aceasta implică **T** *(K)* **£ T**^{2} (K) £ ... £ $\mathbf{T}^m$ *(K) £ ...,* de unde conform consecinţei 1.5.2 există lim $\mathbf{T}^4$ (K) = u **T'** *(K).* Cum operatorul **T** este continuu *n n*
această limită este soluţia ecuaţiei (I.5.8).H

Teorema 1.5.8. *Fie* E_o= **T** (0) u $\mathbf{T^2}$ (0) u ... U $\mathbf{T^4}$ (0) u ... *Atunci* E_n = = **T** (E_o) *şi* E_o £ *E pentru orice E soluţie a ecuaţiei* (1.5.7).

Demonstraţie. Cum 0c **T**(0) şi **T** este izoton urmează T^{4-1}(0) £ $\mathbf{T^4}$ (0). de unde E_o = lim $\mathbf{T^4}$ (0) şi conform teoremei f.5.7, E_o este o soluţie a ecua

ţiei (1.5.7).

Fie *E o* soluţie oarecare a ecuaţiei (1.5.7). Avem 0 £ *E* şi în baza izoto- niei lui **T** urmează $\mathbf{T^4}$ (0) £ $\mathbf{T^4}$ (E) = *E, \/n* E A^7. Trecînd la limită obţinem E_g $E._U$

Această teoremă justifică

Definiţia 1.5.3. *Soluţia minimală* a ecuaţiei (1.5.7) este E_o = ţj **T*** (0). »-i

Dacă E_o = *I** ecuaţia (1.5.6) se zice *ecuaţie universală.*

Teorema 1.5.9. *Ecuaţia* (1.5.7) *are soluţie unică dacă şi numai dacă ea este o ecuaţie univcrsdă sau operatorul* **T** *este nesingular.*

Demonstraţie. Dacă T este un operator nesingular atunci el este contractant şi soluţia este unică conform teoremei de punct fix a lui Banach.

Dacă ecuaţia (1.5.7) este universală pentru orice soluţie *E* avem *I* E* c /• şi deci *E = I*.*

Reciproc, să presupunem că ecuaţia (1.5.7) are soluţie unică. Vom arăta că dacă ecuaţia nu-i universală, atunci cu necesitate operatorul T este nesingular.

Să presupunem că operatorul T ar fi singular, adică T_v / 0. Atunci există un cuvînt *yy... y, k* > 0 care aparţine lui T_v . Deoarece E_o *I**

k

există p E_o .

Fie E = lim T" ({p, e}). Este clar că p E T" ({p, e}), *yn* e *N* şi deci n

p E *E.* De aici urmează că *E* E_o şi că *E* este soluţie a ecuaţiei, ceea ce contrazice unicitatea soluţiei. ■

Observaţia 1.5.6. Rezultatele precedente se extind la ecuaţii de forma :

(1.5.9) **X = T (X)**

unde X E iar T: (^(Z*))"[1] -+ adică X = (Zi, , E_2,..., E_m)

în care E_f, j = 1, m sînt limbaje peste alfabetul finit Z. Operatorul T = (T_x, T_2,..., T_m) va fi în acest caz construit cu ajutorul limbajelor T_k , k = 1, m peste alfabetul Z' = *I* u {i/i,..., y_n},} unde

y, Z, *j* = 1, *n,* astfel : T_t (Ej,... ..., E_n) este limbajul obţinut prin înlocuirea în cuvintele lui T_k a simbolilor *yj* prin cuvintele din limbajele *Ej, j* = 1, *n,* respectiv.

Ecuaţia (1.5.9) se scrie pe componente :

$$E_j = \mathbf{T,}\ (E. \ldots\ldots E_{„}).$$

(1.5.10)

$$E_n = \mathbf{T„}\ (E_l \ldots\ldots E_n).$$

Limbajul T_k se poale scrie prin analogie cu formula (1.5.1) sub formă :

$$k = Z\ U\ T_{k(y}\ u\ 7^{1}{}_{u}$$

unde *1* k_y *= 1 ic* 0 $7_{ki} = T_k$ f) Z*. 7 = 7_k \ { T_{kv} u $T_{k(}$}.

Definiţia operatorului nesingular se extinde în mod natural : T este nesingular dacă T_{kv} =0, \/*k* = 1, *n.* De asemenea rămîne valabilă teorema 1.5.9 pentru sistemul (1.5.10) făcînd schimbările care se impun.

Metrica pe spaţiul $(^3(/^*))^n$ va fi introdusă prin relaţia

$$P\ (Uh\ E'n)'\ (E['E"))\ =\ \max\ \{p\ (E),\ E")\ /\ j\ =\ 1,\ n\}.$$

Dacă componentele Ty ale operatorului T sint operatori conlractanţi. atunci şi T este operator contractant.

§ 1.6. ECUAŢII PE MULŢIMEA LIMBAJELOR

F'ie / = {ij, $i_2z_„$} un alfabet fixat şi /* semigrupul liber generat de *1.* Vom studia aici sistemele de ecuaţii pe echipat atît cu structura de
algebră Boole, cît şi cea de semigrup faţă de produsul limbajelor.

Definiţia 1.6.1. Un *polinom* pe $<p$ (/*) este :

1) un limbaj de forma {i,}, $j = 1$, n şi limbajul vid ;

2) orice limbaj obţinut din limbajele E_t, E_2,..., E_k prin aplicarea de un număr finit de ori a operaţiilor de reuniune, produs şi închidere prin produs.

Numai limbajele obţinute prin procedeele de la punctele 1) şi 2) sînt polinoame.

Definiţia 1.6.2. Un *polinom liniar* în limbajele Z_L . Z_2,..., Z, este un limbaj Z de forma :

$$Z = E_o\ u\ Zj\ U\ ...\ U\ Z_(\ E_t\ ,$$

$\{E_o\ ,\ E_{\pm}E_t\}$ fiind *mulţimea coeficienţilor polinomulni* Z, $E_te\ <p\ (/^*), j$

în studiul unor clase speciale de limbaje intervin sisteme de polinoame de forma :

$$Zj — Zj\ E_{JJ}\ u\ Z_2\ E_{21}\ U\ \bullet\bullet\bullet\ U\ Z_m\ E_{ml}\ u\ Bj \qquad B_2$$

(1.6.1)

$$Z_m \qquad E_{mm}\ \cup\ B_m.$$

Definiţia 1.6.3. O *soluţie* a sistemului (1.6.1) este un m-uplu de limbaje (Zj ,..., Z_m) dincare satisfac sistemul (1.6.1).

Ca şi în algebra liniară folosirea matricilor oferă o modalitate de scriere compactă a sistemelor uşurînd manipularea acestora. în acest scop vom considera matriceale căror elemente sînt

limbaje. Prin $C= (C_{hi}), C_{ht}$ s $<p(I^*). h = 1, n, h = 1, m$ vom nota o astfel de matrice avînd *n* linii şi *m* coloane ; se mai spune că matricea C este de tipul (n, m).

în cazul particular n = 1 matricea va fi numită *veclor-linie,* iar în cazul m = 1, *vector-coloană.* Dacă n *= m* se obţin matricile pătratice.

Fie C = (C_{Alt}), $D = (D_{hlc})$ două matrici de tipul (n, m). Reuniunea lor este matricea C(J $D = (C_{At}|J\ D_{ftJl}.)$ de acelaşi tip.

Dacă C_{At}£ D_{ht}, $h = 1, n, k = 1, m$ atunci vom scrie C s D. Se constată imediat că relaţia „c“ este o relaţie de parţială ordonare în mulţimea matri- cilor de acelaşi tip.

Mai mult, mulţimea matricilor de un anumit tip este semilatice faţă de reuniune avînd ca prim element *matricea nulă (0)* (exerciţiu I).

Fie C = (C_{fcJ}.) o matrice de tip (n, m) şi $D = (D_{tJ})$ o matrice de tip (m, *p). m*

Produsul lor este matricea $CD = (1J\ C_{ftfc}D_{tl})$ de tip (n, p).
fc-i

Cititorul va verifica faptul că mulţimea matricilor pătratice de tip (n, n) formează un *semigrup semilaticial* faţă de operaţiile de produs şi reuniune avînd ca element unitate matricea $U = (U_{j!c})$, unde = {e}, $j = 1, n$ şi $U_{}_{jt} = 0, j / k, j, k = 1$, n.

Închiderea prin produs a unei matrici pătratice *B* este matricea :

$$B^* = \text{U } B^{l}$$

i-o unde $B° = U, B^{i+'} = B^{1}$. *B,* iar

$$= \text{U } B_{iVi}B_{Vi,a}... B_{V}{}_{ik},$$

(V'2.

unde reuniunea se ia după toate j-l-uplele (P_x $^{v}j_i$) care se pot forma
din mulţimea {1, 2,..., n}..

Un monom $B_{U1„2}\ B_{„2„3}$ ... $B_{U(U(+1}$ este :

a) *strict-ciclic* dacă indicii $u_1, u_2, ..., u_t$ sînt distincţi şi $u_{(+1} = u_t$. în particular monomul B_{uu} (deci orice element de pe diagonala principală a unei matrici pătratice) este ciclic.

• -b) *k-ciclic,* dacă conţine *k* monoame strict-ciclice ;

c) *ciclic,* dacă există un *k^-N** astfel încît monomul să fie A-ciclic.

Exemplul 1.6.1. Monomul $B_{13}\ B_{32}\ B_{2l}$ este strict-ciclic, monomul $B_{43}\ B_{32}\ B_{2i}\ B_{44}\ B_{l2}\ B_{25}\ B_{i7}$ *B-, B*м este 3-ciclic, deoarece conţine monoamele strict ciclice $B_{43}\ B_{32}\ B_{2i)}$ şi $B_{12}\ B_a\ B_{i7}\ B_{7l}$.

Teorema 1.6.1. *Orice monom de forma $B_{jh}\ B_{hl}\ B_{h2}... B_{hr\ lt}$ unde B_{jhy},... ...,B_A sînt elemente dintr-o matrice pătratică B, de tip (n,*

n) este cel puftn

+) *ciclic.*

) Am notat cu [x] partea Întreagă a numărului real x

Demonstraţie. Dacă / < /? = 0 şi teorema este banal

Fie deci *r n* şi să presupunem că *In* < r < (/ j 1) n, / e ;V*. Vom proceda prin inducţie după *t.*

Pentru *t* = 1 avem *n r* < 2n şi mulţimea {j, /i± ..... h_{r_l} , A} conţine r + 1 elemente dintre care cel mult *n* < *r* + 1 distincte. Prin urmare, cel puţin un indice apare de două ori şi monomul conţine cel puţin un monom strict ciclic.

Să presupunem acum că teorema este valabilă pentru *l = m,* adică pentru *mn* r < *(m* -ţ- l)n, un monom cu *r* factori esle cel puţin /n-ciclic.

Fie acum (m f 1) *n* < *r* < (m + 2) *n.* Ultimii n factori din monom conţin cel puţin un monom strict ciclic. Partea iniţială rămasă care conţine r — *n* factori, unde *mn* C *r* — *n* < *(m* + 1) *n* este conform cu ipoteza inductivă un monom cel puţin *m* ciclic. Ca atare întregul monom este cel puţin

- ciclic.

Definiţia 1.6.4. O *matrice singulară* esle o matrice pătratică de lip (n, *n). B = (B_{jk})* pentru care există un monom strict ciclic $B_{j}{}_{2}J_{3}... B_{jkii}$ care conţine cuvîntul nul *e* (ceea ce se întîmplă dacă fiecare din factorii monomului conţin pe e) ; indicii j_l , $/_2$,..., j_k se zic *ciclici* în raport cu matricea *B.*

în caz contrar matricea este *nesingulară.*

Observaţia 1.6.1. O matrice pătratică în care cel puţin un element de pe diagonala principală conţine cuvîntul nul este singulară.

Observaţia 1.6.2. Orice putere a unei matrici nesingulare este o matrice nesingulară.

Teorema 1.6.2. *Fie B o matrice nesingulară (le lip (n. n) şi r = mn. m* <s *N*. Atunci :*

min {/ (p) | *p* e $(B^r)_{jk}$. *j,* A = 1, n} = *m.*

Demonstraţie. $(B^r)_{fk}$ este o reuniune de monoaine conţin *r* factori. care

în baza teoremei 1.6.3, este cel puţin tricea *B* fiind nesingulară cuvintele din monoamele strict-ciclice au cel puţin lungimea 1, de unde cuvintele din fiecare monom (deci din $(B^r)_{jk}$) au cel puţin lungimea *m.* ■

Observaţia 1.6.3. Folosind mairicile, sistemul (1.6.1) se scrie sub forma

(1.6.2) *Z = ZE* u *B,*

unde Z şi B sînl vectori linie, iar E este o matrice pătratică : //>!

$_t$ ... F_{lm} |

Z = (Zj ,.... Z_m), ;

«=(Bi..... ~ :

Xe vom ocupa de existenţa şi unicitatea soluţiilor sistemului (1.6.1).

Teorema I.G.3. *Sistemul* (1.6.2) *are soluţie unică Z dacă matricea E este nesingulară şi anume Z = BE*.*

Demonstraţie. Să verificăm mai ini îi că *Z = BE** este o soluţie a sistemului (1.6.2). Avem :

$$ZE \text{ u } B - BE^* E \text{ u } B = B\,(E^* E \text{ u } U) = BE^* = \text{Z}.$$

Să arătăm că aceasta este unica soluţie a sistemului, Fie Z_n o soluţie oarecare a sistemului, adică $Z_n = \text{Z}_o\ E$ u *B.*

Obţinem succesiv

$$\text{Z}_n = (\text{Z}_o\ E \text{ u } B)\ E \text{ u } B = Z_o\ E'^{1} \text{ u } BE \text{ u } B =$$

$$= (\text{Z}_n\ E \text{ u B})\ E^2 \text{ u } BE \text{ U } B = Z_{„}\ E> \text{ U } BE\text{- } \backslash j\ BE\ O\ B =$$

$$\ldots = \text{Z}_{()}\ E^{t+l} \text{ u BE* U ... U } BE \text{ u } B.\ yk = \text{X}.$$

Prin urmare *BE** c Z_o , ceea ce revine la relaţia de incluziune intre elementele corespunzătoare :

(1.6.3) $$(BE^*),\ _{\text{S}}\ (\text{Z}_{„}),\ .\ j = \text{LTT}.$$

Să observăm că produsul a două limbaje conţine cuvîntul nul dacă şi numai dacă fiecare din limbaje conţine cuvîntul nul. Dacă reuniunea a două limbaje nu conţine cuvîntul nul el nu aparţine nici unuia dintre limbaje. Cum *E* este o matrice nesingulară niciunul din limbajele $(\text{E}^*)_{/\text{ft}}$ unde *k* > zn, *./, h* = 1, *m* nu conţin cuvîntul nul.

Fie acum *p* e $(\text{Z}_n)_{;}$ şi *l* (p) = *l.* Alegem *k* + 1 = (/ + 1) *m* şi fie Z_o = = $\text{Z}_o\ \text{E}^{*+1}$ u $BE^{l\,1}$ u ... u *BE* u *B.* Atunci orice cuvînt din orice limbaj nevid $(E^{t+1})_{hj}$, *h* = 1, *m* este de lungime cel puţin *t* -|- 1 în baza teoremei 1.6.2 şi deci *p* neputînd aparţine limbajului $(\text{Z}_o\ E^{*+l})_{)}$ aparţine limbajului (BE*u ... u *B)j ,* dacă (Z_o), g *(BE** >j ... u B)> $(\text{BE}^*)_{\text{y}}$. Avem deci

$$\text{Z}_o = BE^*.$$

şi soluţia sistemului (1.6.2) este unică. ■

Consecinţa 1.6.1. Dacă *e E* atunci ecuaţia Z = *ZF.* u *B* are soluţie unică Z = *BE*.*

Demonstraţie. Rezultă din teorema 1.6.3 pentru *m* = 1. ■

Teorema 1.6.4. *Orice soluţie, a sistemului* (1.6.2) *este de forma* $\text{Z}_{()}$ = (Bu *T)E*. unde T este un vector linie, care satisface la anumite condiţii ce nor fi precizate ulterior.*

Demonstraţie. Fie Z_o o soluţie oarecare a sistemului. Urmind acelaşi raţionament ca în demonstrarea teoremei precedente obţinem *BE** c Z_o . deci

Ca atare $BE^* u\ T = BE^* E$ u $TE\ u\ B = BE^*$ (j $TE,$ de unde TE £ £ BE^* U T şi deci

$$TE^* \text{ s } BE^* \text{ u } TE \text{ £ } BE^* E \text{ u } BE^* \text{ u } T = BE^* \text{ u } T.$$

ş.a.m.d., adică :

$$TE^* \text{ £ } BE^* \text{ u } T, \backslash/k \text{ e N.}$$

ceea ce implică TE^* £ BE^* u $T = Z_o$. Astfel Z_o = Z_o u $TE^* = (Bu\ T)E^*$. ■ *Observaţia' 1.6.4.* Soluţia minimală a sistemului (1.6.2) este $Z = BE^*$. **Teorema 1.6.5.** *Un vector linie de forma* Z_o = *(B* u *T) Fi* este o soluţie a sistemului* (1.6.2) *dacă şi numai dacă T satisface inegalitatea :*

(1.6.4) $$T \text{ £ } BE^* \text{ u } TE^* E.$$

Demonstraţie. Să presupunem că vectorul linie Z_o = *(B* u *T) E** este o soluţie a sistemului (1.6.2). Atunci *(B* U *T) E* = (B u T) E* E* u *B* sau BE^* u $TE^* = BE^*$ u $TE^* E,$ de unde TE^* £ BE^* £ TE^* *E,* adică *T* u U *TE*E £ BE** u *TE* E,* ceea ce implică (1.6.4).

Reciproc, fie *T* un vector linie care satisface condiţia (1.6.4). Vom arăta că Z_o = *(B* u T) *E** este o soluţie a sistemului (1.6.2). într-adevăr, avem Z_0E u $B = (B$ u *T) E** u *B £ BE*U TE* EE* £ BE** u *TE* = (B* u *T)E* = Z_o* şi Z_o = (B u *T) E** £ (B u *BE** u *TE* E) E** = BE* u TE* EE* = B u BE* Eu *TE* E = (BE** u *TE*)* Eu B = [(B u *T)* E*J E u B = Z_o E u B. Astfel Z_o = (B u *T) E** este o soluţie a sistemului. ■

Teorema 1.6.6. *Vectorul-linie T satisface condiţia* (1.6.4) *dacă şi numai dacă* $(T)_f$ *=0 pentru orice j care nu intră ca indice în vreun monom strict ciclic (formal cu elementele matricii E) care conţine cuvîntul nul, (adică pentru orice indice j, neciclic).*

Demonstraţie. Să presupunem că $(T)_j = 0$ pentru indicii *j* neciclici. Atunci, pentru aceşti indici avem evident

$$0 = (T), \text{ £ } (BE^* \text{ u } TE^* E)_j.$$

Dacă *j* nu are proprietatea din enunţ, atunci există un monom strict ciclic care conţine cuvîntul nul *e* şi în care *j* apare ca indice. Fără a restrînge generalitatea putem presupune monomul de forma *Ejhi* ••• (Dacă *j* intra

ca indice într-un monom de forma $E_{lht}\ E_{hiht}E_{htllt}\ldots\ E_{hpf}\ E_{Jhpil}\ldots\ E_{hil}$ am fi putut lua monomul $E_{jh}\ \ldots\ E_h\ ,\ E_{lhl}\ldots\ E_{h\ j}).$ în acest caz avem evident *e* e $(E^*)_n$ şi deci *(T), £ (TE^* ,* A > 1. Cum''TE* £ *BE** u *TE* E,* pentru *k* > 1, urmează $(T)_j$ £ (BE* u *TE* E)_j* . Deci are loc condiţia (1.6.4).

Pentru a demonstra afirmaţia reciprocă este convenabil să utilizăm următoarea „descompunere canonică".

Un vector linie *H,* de tip (1, *m)* se poate descompune în *mod canonic în raport cu matricea E* de tip (m, *m)* sub forma *H* = |J H_z unde :

y _ *((H)*$_{jt}$ dacă *j* este ciclic în raport cu

ţ *0,* în caz contrar,

Ş

(^2)/ 0, dacă *j* este ciclic în raport cu matricea *E (H)ȷ* , în caz contrar.

Fie *T* un vector linie care satisface condiţia (1.6.4) şi Z_o = (B u *T) E* = = BE** u *T* soluţia corespunzătoare a sistemului (1.6.2) existentă conform teoremei precedente.

Fie *T* = 7\ U B U B_2 ’ Z_o = Z_J U descompunerile ca nonice ale vectorilor *T, B* şi Z_o în raport cu matricea *E.*

Avem :

(1.6. Zj c *BE** u 7^1_1 şi Z_2 c *BE** u T_2 .

Plecind de la matricea *E* vom construi două matrici *E, şi* E_2 după cum urmează :

, ,, ,, ,, f $E|_k$, dacă *h* este ciclic, iar *k* nu este

(1.6.6) (Éj)»* = *(0,* in caz contrar,

(E_o, dacă *h* şi *k* nu sînt ciclici

(1.6.7) (^2)** — { .

[0, in caz contrar.

Observăm că matricea E_2 este nesingulară.

Avem Z_L u Z_2 = (Z_1 u *Z2) E* (J *B,* de unde Z_2 = Z_2 E_2 u Z_x E_Y u B_2 . într-adevăr, dacă *j* este indice ciclic $(Z_2)_; = 0,$ iar (Z_2 E_2 u Z_2 E_x u B_2)y = *0* deoarece (Z_2 E_2), = (J $(Z_2)_t$ $(E_2)_w$ = 0, Z_2 E_2 = (J *(Z^k (E^,* = *0* ţinînd k

seama de relaţiile (1.6.7) şi (1.6.6), iar $(B_2)_ȷ$ = 0.

Dacă *j* nu este ciclic atunci $(Z_2)_t$ = Z_t ; = (J $(Z_2)_t$ $(E_2)_w$ u U (^{z}i)t (i)*> *k* *k*

U B_t = (J Z_{tt} E_{tI} u B_f = Z_f, deoarece în expresia u $(Z_2)_t$. $(E_2)_w$ u (J $(Zj)_;$ *k*

(Ej)tj prima reuniune ne dă partea din reuniunea u Z_t *E^* care corespunde indicilor A-neciclici, iar a 2-a, partea corespunzătoare indicilor k-ciclici.

Conform teoremei 1.6.3 matricea E_2 fiind nesingulară, avem :

Z_2 = (Z_2 Ej u B_2) EJ.

Prin urmare Z_o = Z_L u *Z2* = *Z1* u (Z_t Ej u B_2) EJ = Z_2 *(U* u Ej Ej) u . U B_2 EJ . ,

Cum Ej c E. E_2 s E, B_2 £ B avem

Z_o s ZjE* u *BE** = (Z, u B) *E*.*

In baza primei relaţii (1.6.5) găsim :

şi cum avem evident

$$Z_n = (B \text{ u } T)\, E^* (B \text{ u } T,)E'.$$

obţinem

$$Z_o = (B \text{ U } 7\backslash)\ £^*.$$ H

Consecinţa 1.6.2. Dacă *e* s *E* atunci mulţimea soluţiilor ecuaţiei Z - *= ZE* u *B* este mulţimea limbajelor de forma Z = *(B* u *T) E** unde *T* este un limbaj arbitrar.

Observaţia 1.6.5. Teorema 1.6.3 este un caz particular al teoremei 1.6.6 deoarece dacă matricea E este nesingulară nici un indice nu este ciclic cu privire Ia *E.*

§ I. 7. FUNCŢII RECURSIVE

Construcţia clasei funcţiilor recursivc. porneşte de la considerarea unor funcţii numite *funcţii primare.* Aceste din urmă funcţii sini :

1) *funcţia de succesiune* s : *N ->■ N*,* dată de s (n) = n + 1, \/n e A',

2) *funcţia nulă* **o*** *: N*->■* {()}, dată de **o*** (Tj ,.....= 0. V e

A^{7}* . *k* e. A^{7},

3) *funcţia de proiecţie* i* : N^{k} -> A^{7}, dală de i* (ij rj = :t_m. V U'j ..... Tj.) e A'^{k}, 1 *m < k. k. m* e A^{7}.

Vom nota $\mathbf{o^1 = o.}$

Definiţia 1.7.1. *O funcţie numerică parţială* sau pe scurt o *funcţie parţială* este o aplicaţie *f:D-»N.* unde *D* c .V*. Dacă D = N^{k}. *f* este o *funcţie total definită.*

Fie f_1 ..... $f_{„}$ *: D -> 1)*$_{\pm}$, cu *I)* <= A'*, Z), G A şi *g ;])'[* A un număr de *n* -ţ- 1 funcţii parţiale.

Definiţia 1.7.2. *Operatorul de superpoziţie* asupra a *n* -j- 1 *funcţii parţiale.* $\mathbf{S^{B+l}}$ ataşează funcţiilor *g,* f_v .. ., f_n funcţia *f* : D-> A^{7} dată de:

$$f(a|,..., x_{„}) = g(f_x(.Tj..... .r_t),..., f_n(z_l...., x*.)).$$

Vom nota $f = \mathbf{S^{n+1}}(g, f_x ..... f_n).$

Eie *F = {f"' ..... f">}* o mulţime de funcţii parţiale, funcţia *ffi* avind n, argumente, *j* = 1, .*• şi *P* = {i£ | *m, n e N},* mulţimea funcţiilor de proiecţie.

Definiţia 1.7.3. *Mulţimea funcţiilor F- elementare* este mulţimea funcţiilor ce se pot obţine prin operatorul de

superpoziţie din funcţiile din F|J *P.*

P'ie funcţiile parţiale $g : D_s$ -» .V, $h : D_h$ -> N cu D_g c N^n, D_h G N"+'-.

Definiţia 1.7.4. *Operatorul de recursie* asupra funcţiilor *g* şi *h.* **II (</,** *h)* ataşează funcţiilor *g* şi *h* funcţia parţială *f: Df-*N, Df-* N^{n+1} dată de

(1-7.1) /■(.l$_1$ X„.O) = *g(x$_l$ .....*l„).

(1.7.2) /(.r,.....r„ . *y* I- 1) = *h* ţ.r,..r„ . *y. f* (.r,. .*x„ . y)).*

Observaţia 1. 7.1. In cazul II = 0 definiţia funcţiei *f* = II (<■/, /i) revine la :

HO) = «•

f(y r 1) = *h (y, f (y)),* deoarece funcţia *g* se reduce in acest caz la o constantă, notată cu *a.*

Observaţia 1.7.2. Relaţiile (1.7.1) şi (1.7.2) definesc funcţia *f* in mod unic. Avem astfel, pentru $b_k = f$ (.r$_t$ .r„ . *k)* următorul mod de calcul,

$b_a = g$ (.ij ..... x_n).

1 = /' (• 1..... -^{7}n • V

= *h* (*i.....r„ . A- - 1.

Dacă (,r, …r„ , *m) l)$_f$,* atunci ţjj..... x,, . *k)* D$_z$ pentru *k m.* Dacă funcţiile *g* şi *li* sînt peste tot definite, aLunci şi *f* este peste lot definită.

Fie *F* o mulţime de funcţii parţiale :

Definiţia 1.7.5. *Mulţimea funcţiilor F - primitiv recursive* esle mulţimea funcţiilor care pol fi obţinute din funcţiile aparţinîud mulţimii *F* şi din funcţiile primare **|s. o,** /'J prinlr-un număr finit de aplicări ale operatorilor de superpoziţie şi recursie primitivă.

Dacă *F* = 0 funcţiile 0 - primitiv recursive se numesc *funcţii primilii' recursive.* Acestea se pot obţine printr-un număr finit de aplicări ale operatorilor de superpoziţie şi recursie primitivă asupra funcţiilor primare (s, o, P}.

Observafia 1.7.3. Deoarece funcţiile primare sini peste lot definiţe. funcţiile primitiv-recursive sini de asemenea peste lot definite.

Observaţia 1.7.4. Dacă F_t <= P_2 orice funcţie *k*| - primitiv-recursivă este şi F_2 - primitiv - recursivă. De aceea funcţiile primifiv-recufsivc sini *F*- primitiv recursive oricare ar fi *F.*

Exemplul 1.7.1. Conform definiţiei, funcţiile **s. o.** i£ sini funcţii primitiv- recursive. Deoarece*

o" **= S² (o, if).**

şi funcţia o" este primitiv recursivă, $V^{n\,s}$ A^{7}*.

Exemplul 1.7.2. Fie *k : N*^{n} -+ A' o funcţie constantă, *k* (u_{ℓ} ..... u_n) = *a.* Deoarece *k* (u_t..... u„) = **s (s(... s (o"** (ifj //„)), *k este o* funcţie primitiv-recursivă. X

Exemplul 1,1.3. Funcţia *f, (x, y) = x + y* este primitiv-recursivă. intr-adevăr, avem :

/■, (*x,* 0) = x = i} *(x).*

f, (x, y 4- 1) = *h (x, y, f (x,* y)),

unde *h* (x, *y, z) = z* 4- 1. Deoarece *h (x. y. z) =* s (i' (x, *y, :)), h* este o funcţie primitiv-recursivă şi *f* **= R** (i} , *h).*

Exemplul 1.7.4. Funcţia *f* (x, *y) = xy* este primitiv-recursivă

deoarece: /■„ (x, 0) = o (x),

fj, (X< y 4- 1) = *hi (X, y). f_p* (x, *y)).*

unde (x, *y,* z) = *f,* (i? (x, *y, z),* (x, *y,* z)).

Exemplul 1.7.5. Funcţia f_e *(x. y) =* x^{v} este primitiv-recursivă. Astfel :

f. (x, 0) = 1 = *k* (x),

f'(x. y + = h_2 (x, *y,* f_e (x, y)),

unde
h_2 (x, y, *z) =* f_p (i? (x, y, z), ijj (x, y. z)). iar *k* (x) = 1 este o funcţie, constantă. *Exemplul 1.7.6.* Funcţiile:

{ (), dacă x = 0 . — (1, dacă x = 0

şi sg x = {

1, dacă x > 0 (0, dacă x > 0

sînt funcţii primitiv-recursive. deoarece, în baza observaţiei 1,7.1, avem

sg 0 = 0,

sg *(x* 4- 1) = *It (x, sg x).*

tinde *h* (x, y) = 1 este o funcţie constantă. Apoi

$$sgO = 1.$$

$$sg\ (x\ 4\text{-}\ 1) = h\ (x.\ sgx),$$

cu h (x, y) = 0.

Exemplul 1.7.7. Funcţia „diferenţă pozitivă" [$_t$ (x, y) = x — y dată de

$$x \dot{-} y = \begin{cases} x - y, \text{ dacă } x > \\ y \end{cases}$$

este o funcţie primitiv-recursivă. Să arătăm mai întîi că funcţia cp (x) = = x — 1 este o funcţie primitiv-recursivă. Avem

$$cp\ (0) = 0,$$

<? (x 4- 1) = X *(x,* <p (x 4- 1))•

unde x *(x, y)* = i? (x, *y).* Apoi

$$fa\ (x, 0) = i\}\ (x),$$

$$/i\ (*.\ 1/ + 1) - h\ (x, y, f_d\ (x, y)),$$

unde
h (x, y, z) = cp (ij (x, *y,* z)), aceasta deoarece x — *(y* -r- 1) = (x — y) — 1. Ultima egalitate se justifică imediat. Obţinem :

$$1,\ \text{dacă } x > y_\ Jx - y - 1,\ \text{dacă } x > y+$$

$$1\ .\ \text{dacă } x < y \mid \quad 0 \qquad ,\ \text{dacă } x < y4-\ 1$$

$$= x - (y\ 4-\ 1).$$

Exemplul 1.7.8. f_v (x, y) = | x — y | este primitiv-recursivă, deoarece 1 x - y | = (x - y) 4- (y - x) = $f_s(f_d(x, y), f_d(iţ(x, y),$ i?(x, y))).

Funcţia f_m (x, y) = min (x, y) = y — (y — x) este o funcţie primitiv-recursivă, deoarece f_m *(x, y) =* $f_d(il$ (x, y), f_d (iŞ (x, y), ij (x, y))).

Exemplul 1.7.9. Funcţia f_r *(x, y)* dată dc restul împărţirii lui y prin x este o funcţie primitiv-recursivă. Astfel

$$fr\ (x,\ 0) = o\ (x),$$

$$/\bullet_r(x, y\ 4-\ 1) = h\ (x, y, f_r(x,\ y)),$$

unde *h* (x, y, z) = f_p (s (i'(x, y, z)), sg (x, y, z), s (ij (x, y, z))))).

Exemplul 1.7.10. împărţirea întreagă f_t (x, y), funcţia care ataşează perechii (x, y) partea întreagă a citului împărţirii lui y la x este o funcţie primitiv-recursivă, înlrucît

$$f_t\ (x, 0) = o\ (x),$$

$$ft\ (x,\ U + 1) = h\ (x, y, f_t\ (x, y)).$$

unde *h* (x, y, z) = *f,* (i' (x, y, z), sg *(f_e* (i'(x, y, z), **s** *(f_r* (ij (x, y, z), i»(x, y, z)))))). *Observaţia 1.7.5.* Din definiţia funcţiilor primitiv-recursive rezultă că o astfel de funcţie este descrisă de un şir finit de simboli dintr-o mulţime numărabilă. Ca atare, mulţimea funcţiilor primitiv-recursive este numărabilă (v. teorema 1.1.6).

Fie *f: D -» N* cu *D* c N^k o funcţie parţială, pentru care există un procedeu de calcul al valorilor funcţiei. Convenim că (x_L,..., x_{k_l}, x_k) *D* dacă şi numai dacă procedeul considerat nu calculează într-un număr finit de paşi valoarea *f* (x_k..., x_t).

Fixăm primele *k* — 1 argumente şi considerăm ecuaţia

(1.7.3) $$f(x_k.\ldots x_{k_lt}y)=x_k.$$

Pentru a găsi soluţia ecuaţiei (1.7.3), calculăm succesiv, aplicînd procedeul existent, valorile $f(x_j \ldots\ldots\ldots\ldots x_{t_})$. y) pentru y = 0, 1.....

Să notăm cu :

(•.7.4) ..... • *II)* = xj = inin {./ | (/'(x_A $x_{A._t}$ ../) = x_A.}.

Bemarcăin că (f (x_A ,..., x^.^. *y) = x_k)* nu-i definit atunci cind :

1) *f (x, ..., x_{k_t} . 0)* nu-i definii.

2) [(.rj…^{r}*i.-i'j)* este definită şi diferită de ,r_A pentru / = 0. *m* — 1.

iar *f ($x_1x_{t_l}$, rn)* nu-i definită.

3) f (x_A,..., *x_{kl}, j)* este definită şi diferită de x_t pentru orice *j* e A.

în toate celelalte cazuri p *(f* (x_A r^., , *y) = x_k)* este definit şi este dat

de cea mai mică soluţie a ecuaţiei

$$f(X_A \ldots x_{A_}, ../) = x_A. .$$

Observaţia 1.7.6. Atragem atenţia asupra faptului că egalitatea (1.7.4) are loc în mod condiţionat, adică in cazul in care p_y *{f (x_1 ..... .$r_{A._1}$. y)* = x_A) este definit. Astfel *(f* (x_t . *y)* = .r_2). unde *f* (x, . x_2) = x_2 (x_2 — (x, 1)) nu-i definită, deoarece *f* (.r, . 0) nu-i definită. Pe de altă parte avem:

min *{j* I *f* (Xp ./) = 0} = .r, ■ 1. iar egalitatea (1.7.4) nu arc loc.

Definiţia 1.7.6. *Operatorul de minimi zare* M ataşează funcţiei *f* funcţia parţială VI (f): -> .V dată de :

M (/) (< j.»'*)= *(f* (-' i.....M = *P-Af* (' i. .> '/)= •«'»)• -^{v}-

Observaţia 1.7.7. Pentru A = l. funcţia parţială VI (f) este funcţia inversă f^1.

Exemplul 1.7.11. Fie funcţia /", (x_A , x_2) x ; x_2. Avem

VI (f,) (x, . x,) = a_y (x, v *y* = x_2) ■ ' 2 ~ ' 1 ■ ^{uiu}

= este dată

Exemplul 1.7.12.

Funcţie .VI (sg)

VI (sg) (x) = x, dacă x < 1, nedefinită, dacă x> 1.

iar .VI (s) se obţine din :

Observaţia 1.7.8. Numerele: **VI (s) (x) =** x — 1, dacă x > 0 nedefinilă. dacă

•d_v(i *f* (-^{r}i..*y)* -.9 (fi • . *y)*| = 0.

F„(Sg | *f* (Xj x_t • .'/) - .9 (Ti , y)| = 1).

sint de fapt respectiv

[x_y *(f* (Xj *x_k, y) = g* (X] …r_A.. *y))*

Definiţia 1.7.7. O *funcţie F - parţial recursivă* este o funcţie parţială, care se obţine clin funcţiile din familia F şi din funcţiile primare **s,o, i£** prin- Ir-un număr finit de aplicări ale operatorilor de superpoziţie. recursie primitivă şi minimizare.

O funcţie 0 - parţial recursivă se numeşte *funcţie parţial-recursivă.*

Observaţia I.7.U. Orice' funcţie *F-* primitiv recursivă este şi F-parţial recursivă. în particular, o .funcţie primitiv-recursivă este şi parţial-recursivă.

Observaţia 1.7.10. Clasa funcţiilor parţial-recursive include strict clasa funcţiilor primitiv-recursive, deoarece printre funcţiile parţial-recursive se află funcţii parţial definite (v. exemplul 1.7.10), în timp ce funcţiile primitiv- recursive sînt peste tot definite (v. observaţia 1.7.3).

Fie *AI* £ N. Vom ataşa mulţimii *AI* două funcţii numerice şi anume funcţia total definită :

$$(n) = \begin{cases} 0, \text{ dacă /i} \\ \text{s .17,} \end{cases}$$

şi funcţia parţial definită :

$$l>^{P}M\,(\hat{}) = \begin{cases} 0, \text{ dacă } n \text{ e .17, nedefinită, în caz} \\ \text{contrar.} \end{cases}$$

Observaţia 1.7.11. Avem h_M (n) = 1 — (n). Apoi, dacă .17 = 0, atunci
h_M (n) = 1, Vzi s .17, iar este peste tot nedefinită.

Definiţia 1.7.8. O mulţime .17 £ .V este *primitiv-recursivă (parţial-recursivă)* dacă funcţia h_M este primitiv-recursivă (dacă /ij) este parţial-recursivă).

Observaţia 1.7.12. Orice mulţime primitiv-recursivă este parţial-recursivă, deoarece $h\hat{}(n) = 0 — h_M(n)$.

Observaţia 1.7.13. Funcţia numerică parţială:

$$\mathrm{p}_{(/1)} = \begin{cases} \text{I } f\ W' \text{ (la(ă " } s \\ \text{nedefinită dacă } n \text{ ţ£A7,} \end{cases}$$

este parţial recursivă, unde *f* este o funcţie primitiv-recursivă, iar .17 £ .V o mulţime primitiv-recursivă. într-adevăr, funcţia este o funcţie parţial recursivă (v. observaţia 1.7.12), iar f^{p} (n) = f (n) + $h[_t$ (n).

Observaţia 1.7.14. Următorul enunţ este cunoscut sub numele de *leza lui Alonso Chureh : „Clasa funcţiilor numerice pentru care există algoritmi de calcul coincide cu clasa funcţiilor parţial-recursive".*

Teorema 1.7.1. *Fie f: D -» N, D £ N" o funcţie primitiv*

(parţial) recursivă. Atunci funcţia g: N, D_t £ N^a definită prin :

(1-7.5) $$g(-U,\ldots,x_H) \overset{x_n}{=} \hat{}Jf(x_l,\ldots,x_{n_L},j)$$

este de asemenea primitiv-recursivă (parţial-recursivă).

Demonstraţie. Din egalitatea (1.7.5) obţinem :

$$9(*1\ X_{n_v}\ 0) = f(\text{Xj},\ldots,X_{B_X},0),$$

$$g(.?!x_{B-1}, y+1) = g(\text{xj}\ldots..x_{B_x}, y) + f\ fa,\ldots,\quad ,y+1)$$

şi deci funcţia *g* se obţine prin recursie primitivă din funcţiile recursive / (a*i....., 0), *h fa x_{a_L}, y, z)* = z+/'(x_1,..., $T_{B_}$! , *y*+ 1) şi de aceea *g*
este primitiv (parţial) recursivă.H

Teorema 1.7.2. *Dacă funcţia f:N*->N este primitiv recursivă, atunci funcţia g : N* -» N dată prin*

(1-7.6) $$g\ fax_{\text{„}}) = \text{JJ}\ f(x_x,\ldots,x_{n_r},J),$$ $$l\text{-}o$$

este de asemenea primitiv-recursivă.

Demonstraţie. Se procedează ca în teore ma precedentă cu *h fa..... x_{n_v} y,* z) = *zffa x_{n_v} y* + 1). ■

Teorema 1.7.3. *Fie g* : A^{TB+1} ->■ *N* şi *OL: N^a ->■ N două funcţii primitiv-recur- sive, astfel incit ecuaţia g (x_Yx_n , y)* =0 *să aibă cel puţin o soluţie in y, oricare ar fi fa,...,* x„) e N^n *şi*

(1.7.7.) p_y *(g fa,...,* x_n , *y)* = 0) oc (Xj ..... x„), *V fa .....* x_B) e *N"*

Atunci funcţia f:N^m->- N dală de f (x_t,..., x_B) = p_y *(g fa,...,* x„) = 0) *esle de asemenea primitiv-recursivă.*

Demonstraţie. Fie (Xj,..., x_n) e *N"* şi *a* = p_y *(g* (Xj,..., x_B , *if)* = 0). Con- *1*
siderăm şirul *g* (Xj..... r_B , *k), j* E A^7. Din definiţia lui *a* rezultă că
k-0

(1.7.7) TZ_0 ,.... , sînt nenuli şi $7z_{o+B>}$ = 0, Vme Ai. Astfel, din condiţia obţinem *a* = V) {sg $7t_{,}$ | *j* = 0, a fa,..., x_B)}. Introducînd funcţia primiliv-re- z
cursivă /i (x_t,..., .r_B , z) = JJ *g fa ..... x_n .j)* ultima egalitate se scrie sub *i-0*
forma :

$$f\ \textbf{(*1}\ {}^{X}n) = \textbf{S\ \{\ Sg}\ h\ \textbf{(Xi ,..., X.}\ , J)\ |\ j = 0,0\ \textbf{(Xj ,..., xj\}.}$$

Afirmăm că *f* este o funcţie primitiv-recursivă. într-adevăr, sg *h fa...* .... x„ , z) este o funcţie primitiv-recursivă în baza exemplului 1.7.6. Apoi, în baza teoremei 1.7.1, funcţia *t fa ,...,*

x_u , *v)* = sg A (x_t,..., x_B , j) estepri- *i-o*
mitiv-recursivă şi *f* (Xj,..., $x_{„}$) = *t fa,..., x_n. o fa* ,..., x_B)). ■

Consecinţa 1.7.1. Funcţia [-^x] este primitiv-recursivă. într-adevăr, egalitatea n = [/x] este echivalentă cu n^2 < x<^(n + l)^2. Astfel [Vx] = p; (sg ((/ + l)^2 — x) = l)<x şi se foloseşte teorema precedentă.

Observata I.l.lo. Am arătat (v. exemplul 1.1.11) că mulţimea A^7* x A'* esle numărabilă. Cum A' x N = (A^7* x A'*) U {((). ())} u ({0} x .V*) u U (A's x {0}), rezultă că şi mulţimea A^7 x N este numărabilă. Fie A^7 *x N* = = {(0,0), (0,1), (1,0), (0,2), (1,1), (2,0),... e (x, y) locul ocupat de perechea (x, *y)* în şirul de mai sus şi 1 (n) = x, r (n) = *y,* dacă e (x, *y) = n.*

Avem

$$+ 1]+1 \quad \mathbf{e}(x, y) = \frac{(x + y)(x + y}{2}$$

Acest operator se defineşle pentru orice *i* s *I.*

Operatorul <le i-minimizare Si, ataşează funcţiei *f: D -» I** funcţia parţială M (f) : Bj ->■ *I** dată de :

$$\text{M (f) (Pi}\mathit{Pic)} = [f\,(Pi —, Pn\text{-}i\,, y) = 4$$

unde *y* E {()*, iar *{f* (p, ,..., p_{lc_l} . *y)* = c] = i^n cu *ii* = min |/n j *f* (p_l ....

$$..., p_{k_l}\,, i \ldots i = \text{e}\},$$

iar (p_x,..., , i^h) e D_f, 1 < *h* < /;.

Noţiunile pe care le prezentăm în continuare joacă un rol important in studiul unor limbaje (v. § 11.12 — 14).

Definiţiile 1.7.5 şi 1.7.7 se extind corespunzător la funcţii definite pe semigrupuri libere.

Să presupunem că | *1* / = *k. 1* = i_L, i_2,..., i_k . Cuvîntului p = i_h i_{jt}... i_{jm} i se ataşează numărul natural *n* = (j\ — 1) + (j_2 — 1) *k* + ... + (j_{OT} —1) k^m scris în baza *k.* Corespondenţa astfel stabilită este o bijecţie între semigrupul liber *1** şi mulţimea numerelor naturale.

Vom putea scrie de aceea, în continuare, în locul cuvîntului p numărul natural n, care îi corespunde prin procedeul de mai sus.

Definiţia 1.7.11. O *mulţime recursivă E £ 1** este o subinulţime a lui I*, a cărei funcţie caracteristică :

$$'/E\,(") = \begin{cases} [1, \text{ dacă } n \text{ e } E \\ | \ 0, \text{ dacă n 7-.'}, \end{cases}$$

esle o funcţie recursivă.

Definiţia 1.7.12. O *mulţime recursiv enumerabilă E £ I** esle o subinul- ţime a lui *I*,* pentru care există o funcţie recursivă, total definită, *f: I* -> /*, astfel îneît

$$E = \{\text{f (n)} \mid \text{zi } 7^*\}, \text{ sau } E = 0.$$

Observaţia L7.16. Deoarece 7* este o mulţime numărabilă, orice parte 7: £ 7* este o mulţime numărabilă. Dacă *E* este, în

plus, recursiv enumera- bilă, aceasta înseamnă că există o funcţie recursivă ale cărei valori sînt cuvintele din *E.* Astfel, pentru un anume cuvînt p s 7* (căruia îi corespunde numărul *n),* în cazul în care p s *E,* construind succesiv şirul *f* (0), *f* (1),... vom ajunge, după un număr finit de paşi, să avem *f (m) = n.* Vom putea astfel rezolva problema apartenenţei cuvîntului p la *E.* Deoarece în baza tezei lui Church există un algoritm pentru calculul funcţiei *f,* rezultă că o mulţime *E* este recursiv enumerabilă, dacă şi numai dacă există un algoritm care permite alcătuirea unei „lisle" a cuvintelor din *E.*

Teorema 1.7.4. *O mulţime E* £ 7* *este recursiv enumerabilă, dacă şi numai dacă există o funcţie recursivă, total definită, de două variabile, y : I** x *I** -» 7*, *astfel incit* 3.r e 7*, *y (a, x)* = 0, *atunci şi numai atunci cind a* s *E.*

Demonstraţie. Fie *E c I o* mulţime recursiv enumerabilă, *E* / 0. Există atunci o funcţie recursivă, total definită *f: I*-* I** cu *E = {f ţn)* | n s /*}. Fie atunci funcţia *h* : /* x *I** *I** dală de :

$$/'(/>.\ 7)=|,,\ \begin{cases} 0, \text{ dacă } p = a \\ \text{ţ } 1, \text{ daca p / } q \end{cases} = sg«P <l)^{v},\ (q\sim l>))\text{-}$$

Se constată imediat că *h* este o funcţie recursivă. total definită. Funcţia *g ţp, q) = h ţp,' f* (<?)) joacă rolul din enunţ.

Invers, să presupunem că pentru mulţimea *E* c /* există o funcţie *g* cu proprietatea cerută, adică *E* este mulţimea acelor cuvinte *a* s *I** pentru care ecuaţia *g ţa, x)* - 0 este rezolvabilă.

Considerăm funcţia primitiv recursivă *k ţn)* = 1 (n) sg *g* (l(n), **r (/)) +** + & ^{S}S *9* (1(0, ^{r} (0), unde 1 (/) şi **r (/)** sînt funcţiile introduse în observaţia 1.7.15, iar *b* s *E.* Afirmăm că E = {k (n) | n s /*}. Intr-adevăr, dacă pentru un *n* e *I** avem *g* (l(n), *r ţn)) =* 0, atunci 1 (n) s *E* şi, conform definiţiei lui k, avem k (/) = l(î) s *E,* iar dacă *g* (l(n), r (n)) / 0, atunci k (/) = *b* e *E.* Astfel *E* 2 {k (n) j n s /*'. Fie acum *a e E.* Atunci există xe/* cu *g (a, x)* = 0. Punînd *n* = e (a, x) şi, deoarece 1 (c (a, x)) = *a,* obţinem k (n)= *a,* deci *E* G *{k* (n)|n e /*}•■

Consecinţa El.2. Fie *G ţa,* ,..., x_B) o funcţie primitiv-recursivă. Mulţimea acelor valori ale parametrului *a* pentru care ecuaţia *G ţa,* Xj,..., x_B) = 0 are cel puţin o soluţie (Xj,..., x„), este recursiv enumerabilă. Să introducem funcţia primitiv recursivă *g ţa,* x) = *G ţa,* **c_{B1}** (x) ,..., e_{BB}(x)), unde e_{ti} sînt funcţiile introduse în observaţia 1.7.15. Deoarece *g ţa,* **c"** (x_x,..., x„)) = = *G ţa,* X!,..., x_n), urmează că mulţimea valorilor parametrului *a,* pentru care ecuaţia *g ţa,* x) = 0 admite o soluţie în x, coincide cu mulţimea acelor *a* pentru care ecuaţia *G ţa,* x_x,..., x_B) = 0 are soluţii în x_x,...,

x_B .

Teorema 1.7.5. *Reuniunile şi intersecţiile finite de mulţimi enumerabile sînt mulţimi enumerabile.*

Demonstraţie. Fie E_t,..., E_n c /*, *n* mulţimi recursiv enumerabile şi *ţa,* x),.„ g_a *ţa,* x) funcţiile primitiv recursive ataşate acestor mulţimi conform teoremei 1.7.4. Atunci totalitatea valorilor parametrului *a* pentru care ecuaţiile g_t *ţa,* x_x)... g_n *ţa,* x„) = 0 şi g_r *ţa,* x>) + ... + $g_„$ *ţa,* x„) = 0 coincid respectiv cu A = u {$E_;$ | j = 1, n} şi B = n {$E_;$ |j = 1, n}. în baza consecinţei 1.7.2, mulţimile A şi B sînt recursiv enumerabile.

Teorema I.7.G. *O mulţime recursivă este recursiv enumerabilă.*

Demonstraţie. Fie E o mulţime recursivă. Atunci funcţia ei caracteristică X_B este o funcţie primitiv recursivă şi ecuaţia 1 — («) + x = 0 are o so
luţie în x (x = 0) atunci şi numai atunci cînd *a* **e** E.■

Reciproca acestei teoreme ne este adevărată, cum se va arăta în § 11.14 (v. teorema II.14.6). Totuşi, are loc :

Teorema 1.7.7. *Dacă mulţimile* E c /* *şi* E^c = *I** \ E *sînt recursiv enumerabile, atunci E şi* E^c *sînt recursive.*

Demonstraţie. Fie *g* şi *h* două funcţii primitiv recursive cu E = {a | *a* s /*, gxeZ*, *g* (u, x)= 0}, E^c = *{a \ a I*, 3x<^ I*, h (a,* ,r) = 0}. Deoarece pentru orice *a I** una din ecuaţiile *g* (a, a) = 0 sau *li* (a. a) = 0 este satisfăcută funcţia recursivă

$$f(«) = Fx\ (!)\ («, •<)\ h\ (<t, •?) = 0)$$

este peste Lot definită. Prin urinare, funcţia G (a) = 1 — sg *g (a, [* (a)) este de asemenea o funcţie recursivă, peste tot definită. Dacă G («) = 1, avem sg *g* (a, *f* (a)) = 0 sau *g (a, f* («)) = 0, de unde *a e E.*

Invers, fie *a* s *E.* Atunci ecuaţia *g (a, x)* = 0 are o soluţie. Cum *a* E^c avem *h (a,* a) / 0, Va e= /* şi deci *(g* (a, a) = 0) = *',i_z (g (a,* a) *h* (a, a) = = 0) = *f (a).* Astfel *g (a, f(a))* = 0 şi G (a) = 1.

Funcţia H (a) = sg *g* (a, *f* (a)) este funcţia caracteristică pentru mulţimea E^c. /

§ I. 8. ELEMENTE DE LOGICĂ MATEMATICĂ

Ne vom ocupa mai întîi de construcţia axiomatică a calculului

propoziţiilor, instrument care permite algebrizarea unei importante părţi a activităţii de gîndire.

Definiţia 1.8.1. *Simbolii de bază ai calculului propoziţional* sînt :

i) *simbolii variabilelor propoziţionale* {u_B | *n* s JV}

ii) *conectivele logice* ~| şi A (care se citesc *„nu“* şi ,,şi“)

iii) *semnele de punctuaţie* (şi) (paranteza stingă şi dreaptă)

Vom nota cu l_{prop} mulţimea simbolilor de bază ai calculului propoziţiilor şi vom pune în evidenţă limbajul furmulelor calculului propoziţional F_{prop} c/*$_{rop}$

Definiţia 1.8.2. *Limbajul formulelor* este dat de următoarele reguli :

*) I ^{n} — ^*prop

ii) Dacă a, [3 e F_{prop} atunci Joc, (a A P) ^{e} ■

iii) Un cuvînt dinZ^*_{pr0P} este o formulă dacă şi numai dacă se obţine prin- tr-un număr finit de aplicări ale regulilor de mai sus.

Observaţia 1.8.1. F_{prop}este cel mai mic limbaj care conţine variabilele propoziţionale şi care, odată cu cuvintele a şi p conţine cuvintele "|a şi (a A P)-

Exemplul 1.8.1. Următoarele şiruri de simboluri sînt formule : "jx, 1 (« Al P). 1 (1 (« A P) A 1 (l(a A P) A 1 P))-

Observaţia 1.8.2. Vom introduce trei prescurtări pentru formule din F_{prop}. Anume, vom nota (a V P) în loc de 1("|a A 1 P)> ^{cu} (^{a} P) $^{vom\ nota}$ l(^{a} A 1P) ŞÎ> în fine *^{n} loc d^{e} (($^{a->}$ P) A (P ■* °0) ^{vom} P^{u}ne (a«-» (3).

Din punctul de vedere al calculului propoziţional ne interesează numai dacă o anume propoziţie este adevărată sau falsă.

Definiţia 1.8.3. *O realizare a calculului propoziţiilor* este o aplicaţie *f:* , *n* e N} -> B_2, unde B_2 este algebra Boole cu două elemente, B_2 —
= {o.i.u.n.-}-

Observaţia 1.8.3. O realizare a calculului propoziţiilor poate fi extinsă la o aplicaţie *f*: F_{prop} -> B_2 punînd în mod recursiv :

(I.8.D f (a/\ p) =/•(«) A f (?)

(1.8. f(» = Ha)-

De aici rezultă că :

(1.8.3) Ha V P) = /•(ICI « A 1 0)) = (f(«) A f(0)) = *f* («)U H0)- unde semnele „Q" şi „(J" se referă la operaţiile de intersecţie şi reuniune din algebra Boole.

Definiţia 1.8.4. *Formula ct satisface realizarea f dacă f* (a) = 1. Vom nota aceasta prin *f*|= a.

Formula a este o *tautologie* dacă satisface *orice* realizare *f.* Vom nota cu **Taut** mulţimea tautologiilor.

Observaţia 1.8.4. Faptul că *f* (a) = 1 semnifică în mod intuitiv că formula a este adevărată pentru realizarea *f.* în caz contrar a se interpretează ca falsă pentru această realizare. Astfel, de exemplu formula a /\ p este adevărată pentru realizarea *f* dacă şi numai dacă atît a cît şi p sînt adevărate pentru această realizare (v. formulele (1.8.1)).

Observaţia 1.8. j. Dacă formula a este construită cu ajutorul variabilelor $v_Y v_n$ şi *f (Vj) = g (vj), j* = 1, *n* atunci *f* (a) = *g* (a). Astfel formula a determină funcţia booleană f_a : BJ B_2, unde f_a $(t_v,..., t_n) = f$ (a), *f* fiind realizarea care aplică v_j pe elementul t_j *e* B_2.

Rezultă de aici că a este o tautologie dacă şi numai dacă f_a $ţt_l ,..., t_n)$= 1, V (tj/„) s B" , de unde urmează că este decidabil[1]) dacă o anume formulă este o tautologie.

Definiţia 1.8.5. *Axiomele calculului propoziţional* sînt următoarele formule :

i) a-» (P-> a)

ii) [a -» (P -> y)J - [(a P) (a -» y)]

iii) C|a-*iP)-*(P->a)

iv_a) (a A P) -> a

iv_6) (a A 0) -> P

v) [y a] ->■ [(y -> 0) (y [a A 01)1

vi») a -> (a V 0)

[vi]o) 0 -> (a V 0)

vii) [a -> y] [(0 -> y) -> ([a V 01) î)]

***) o problemă este** *decidabili* **dacă există un algoritm pentru rezolvarea ei.**

Observaţia I.S.6. Axiomele calculului prepoziţional sini tautologii. De exemplu, fie *f* o realizare a calculului propoziţiilor. Avem

/>• - - a)) = /-(a) O *f* (3 - a) = f (a) Q (f(P) Q [(?.) =

= n*) nnp) rW) = 0 = 1 eic.

Definiţia 1.8.6. *Formula^ esle o eonseeinţă imediată a* formulei a *şi* a -> [3, dacă a, a-> [3 s $F_{PI0I>}$. Acest procedeu de a deduce consecinţe imediate din formule poartă numele de regula *„modus ponens".*

Definiţia 1.8.7. *0 demonstrare a linei formule* a este un şir finit de formule a, a„ astfel incit a„ = a şi *v.*$_t$ este sau o axiomă sau există *j, li* </; astfel incit a$_{Ă}$. să fie o consecinţă imediată a formulelor şi a$_A$, V/>' = 1, *n* ; numărul *n* este *lunyimea demonstraţiei.*

Formula a esle *inferabilă* dacă există o demonstraţie a ei. Vom nota acest lucru prin I-a iar cu Dom vom nota mulţimea formulelor inferabile.

Teorema 1.8.1. *Avem* **h** a -» a, Va e F_{PIOV}.

Demonstraţie. Fie axioma ii) pentru a = y şi p = (a -» a) :

[a ([a -> a] -» a)] ->■ [(a -> [a a]) -> (a -> a)].

în baza axiomei i) putem scrie tautologia :

a -> [(a -> a) -> a] ,

unde am luat p = (a—► a)

Rezultă. în baza regulii modus ponens tautologia :

(a -» [a -» a]) -» (a -> a).

Apoi, luînd în i) p = a găsim a-> [a -» a], de unde aplicînd din nou regula modus ponens avem

a -> a. ■

Lăsăm în seama cititorului demonstrarea faptului că următoarele formule :

(» -* P) -* [(y - a) (y -* P)]

a - (a - p)

(a A P) - (P A a)

« -> [P (« P)1

[(a A Y) V (P A Y)1 -* [(a V P) A Y]

(a -» 8) -* [(P ->■ (Y -> a)) -> (P -» (Y "* S))l

sînt tautologii.

Teorema 1.8.2. *Avem* **Dem £ Taut.**

Demonstraţie. Voiri proceda prin inducţie asupra lungimii demonstraţiei. Pentru n — 1 faptul rezultă din observaţia 1.8.3.

Să presupunem acum că a şi a-» p sînt tautologii şi să arătăm că şi Ş este o tautologie (modus ponens).
Avem *f* (a) = *f* (a -> P) = 1. de unde

= [/■(*) nfwiu HP) = [/■(«)u /(p)j n

n [/■(*) n /■(?)] = ti u Mp)] n /■(« - ?) = 1 n 1 = 1. ■

Consecinţa 1.8.1. Nu există formulă 7. astfel incit atîl 7. cil şi la să fie infera bile.

Intr-adevăr, *f* (a) = 1 dacă şi numai dacă *f* ("la) = 0 şi de aceea a şi "|a nu pot fi simultan tautologii. Cum **Dem £ Taut** rezultă că a şi "|a nu pot aparţine simultan cu **Dem.**

Observaţia 1.8.7. Relaţia ,,-<"pe F_{prop} dată prin: a-< p dacă şi numai dacă 1— (a —* P) este o relaţie de preordine (reflexivă şi tranzitivă). Intr-adevăr, reflexitalea ei rezultă din teorema 1.9.1, iar tranzitivitatea se obţine remarcînd că din I- (a -»■ p) rezultă *f* (a) < *f* (P), oricare ar fi realizarea *f.* în schimb, această relaţie nu este antisimetrică, deoarece, dacă a, *b* e F_{prop} sint două formule distincte putem arăta că a A P -< P A [a] Ş> P A [a] ~< [a] *A* P deşi a A P Ş* P A [a] sînt în mod evident formule distincte.

Observaţia 1.8.8. Relaţia = £ F_{prop} x F_{prop} dată prin a = p dacă şi numai dacă I— (a-> p) şi |- (p~> a) este o echivalenţă. Vom nota cu [a] clasa de echi- valenţă a formulei a. Relaţia „=" este dată de „=" = Q (ker *f* / *fe* B_2^{vr011})-

Observaţia 1.8.9. Relaţia sg £ F_{prop}/= x F_{vmv}*l* = dată de [a] < [pjdacă şi numai dacă I- (a -» p) este o relaţie de ordine parţială pe F_{prop} / = . Corectitudinea acestei definiri rezultă din faptul că dacă I- (a -> P), a = a', P= P' avem |_ (a' -» P').

Avem [a] sg [p] dacă si numai dacă pentru orice realizare *f* găsim Ha) < f(P).

Teorema 1.8.3. *Mulţimea parţial ordonată* (F_{prop} / = , =g) *este o algebră Boole şi* [a] = 1 *dacă şi numai dacă* I- a, *iar* [p] = 0 *dacă*

şi numai dacă I— “] p.

Demonstraţie. Fie *a,* p s F_{prop}. Să arătăm că sup {[a], [p]} = [a V PL inf (O-L (Pl} = [« A Pl

in baza axiomei vi_a), avem [a] [a V PL [Pl < [^{a} V PL Dacă [a] < [YL [Pl < M ^{avem} ?(^{a}) < HYL HP) < HY) ŞL conform cu 1.9.3 obţinem *f* (a y p) < *f* (Y), V *f* « Bfprop ceea ce implică [a V Pl < [ŸL Similar se procedează pentru infimum.

Distributivitatea rezultă din tautologiile :

1- [(« A p) V (Y A P)1 [(* V y) A PL

I- [(a $_V$ p) A p] - [(« A P) V (Y A P)L

valabile pentru orice a, p, y e F_{prop} (exerciţiu !)

Dacă Ha atunci f(a) = 1 şi pentru orice S E F_{prop} avem *f* (S) < *f* (a), de unde [a] este cel mai mare element al laticii F_{prop} / =.

Similar se arată că dacă |— "| Ş atunci [fl] esle cel mai mic element al laticii /^prop / == •

în sfîrşit, deoarece

Ha V şi H "|(a /\ Ja)

(v. observaţia 1.9.2 şi teorema 1.9.1) găsim [a] V [1 a] = 1 şi [a] A ["| a] = 0, ceea ce indică faptul că F_{prop} / = este o algebră Boole. ■

Această algebră este cunoscută în literatură (v. Beli, J. L. ; Slomson, A. B. [1]) sub denumirea de *algebră Lindenbaum.*

Următorul anunţ este cunoscut sub numele de *teorema de completitudine a calculului propoziţional.*

Teorema 1.8.4. *Avem* **Dem = Taut.**

Demonstrare. Se va consulta [Beli, J. L.; Slomson A. B. [1], p. 43].

Definiţia 1.8.8. Alfabetul simbolilor de bază ai calculului predicatelor *Ipn<i* este mulţimea alcătuită din :

i) *simbolii variabilelor individuale | n* s N}.
ii) *simbolii predicativi {P_n* | n E .V]
iii) *simbolul egalităţii =*
iv) conectivele logice “] şi A
v) cuantificatorul existenţial 3
vi) simbolii de punctuaţie ( ).

Vom considera *aplicaţia de graduare g :* {P_B , *n* E N] -» *N,* iar *g (*P_n) va fi numit gradul simbolului predicativ P_n. Pentru simplitate *g* (P_B) va fi notat cu *g* (n).

Simbolii logici sînt {"], A, 3}.

Definiţia 1.8.8. O *formulă atomică* este un şir de simboli de forma *x = y* sau de forma P_n (Xj,..., $x_{e(n)}$).

Definiţia 1.8.9. *Limbajul L al formulelor calculului*

predicatelor este dat de următoarele reguli :

i) Formulele atomice fac parte dinL

ii) Dacă <p, E *L* şi *v* este o variabilă atunci 1<p, (cp A '[')> (3f)ţi^{G} *L*

iii) Un şir de simboli de bază este o formulă dacă şi numai dacă se obţine printr-un număr finit de aplicări ale regulilor de mai sus.

Observaţia 1.8.10. Limbajul *L* al formulelor calculului predicatelor este cea mai mică mulţime de cuvinte din semigrupul liber Z^*_{pied} care conţine toate formulele atomice şi care, împreună cu cp şi tp conţine pe 1<p, (<p A^) Şi (3^)<p. oricare ar fi simbolul de variabilă *v.*

Observaţia 1.8.11. Vom folosi şi în calculul predicatelor simbolii /\, —» cu aceleaşi semnificaţii ca şi în calculul propoziţiilor (v. observaţia 1.8.2). în plus formula ") (3x) I9 va fi notată cu (Vx) 9.

Este clar că dacă 9, E *L* atunci (9 y tp), (9 -* <p), (9»-»+) şi (Vx) 9 sînt formule.

Definiţia 1.8.10. *Un v-cuantificator* este un şir de forma (Sa). Domeniul unui u-cuantificator se defineşte recursiv astfel :

i) Domeniul lui (3a) în formula (3a) 9 este (3a) cp.

ii) Domeniul lui (Sa) în ”|«p (Sa) este acelaşi ca în <p (Sa)

iii) Domeniul lui (Sa) în 9 (Sa) AZ ^{sau} zA'P(^a)^ este acelaşi ca şi în 9 (Sa) i|>.

iv) Domeniul lui (Sa) în (3i/)©(3a)^ esle acelaşi ca şijîn formula ?(3a)<p. *Exemplul 1.8.2.* Să considerăm formula "| ($3a_0$) [(SaJ [(Sa_2)P ($a_{,,}$ a_2).

P ("1 ¸ i'o)] V 1 P ("o > "o)L Domeniul cuantificatorului (Sa_x) este (Sa_t) [(Sa_2) *P (i>$_L$,* a_2) /\ *P* (aj, a_0)J.

Definiţia 1.8.11. O *apariţie* a unei variabile a într-o formulă este o *apariţie legală* dacă ea apare în domeniul unui a-cuantificator. în caz contrar apariţia lui a este *liberă.* în aceeaşi formulă o variabilă poate apare şi liber şi legat.

Dacă în formula 9 variabilele $x_{\pm}$, ... x_n sînt libere vom nota acest lucru cu o (rq $x_{„}$).

O formulă *deschisă* esle o formulă care nu conţine cuantificatori. închiderea formulei 9 (2^ x_n) este formula (Vi'J ... (Vx_n) 9 (x_Y ...., x_n).

Definiţia 1.8.12. O *propoziţie* este o formulă în care nu avem apariţii libere ale variabilelor.

Definiţia 1.8.13. *Axiomele calculului predicatelor* sînt

A. i) 9 —> (<p —> 9)

ii) [? (P -* z)l -> ([? -* <P) -* (? -*■ z)l

iii) (19 1+) (+ -* ?)

iv„) (9 A <P) -> ?

iv„) (9 $_{A}$,p) ţp

v) (z -> ?) -* [(z-* +) *(7.* -* [? A 'Pl)l

vi.) © -> (9 V p)

vi$_{6}$) 'P (9 V Ip)

vii) (<? z) [(<Hz) -> (l? V<P1 -* z)l

B. i) (Vx) *<p (x)-> <p (y)*

ii) *(V.r)* (9 -> <P) (9 (Vx)ip)

C. i) (V.r) *(x — x)*

ii) *(yx)(yy)(x = y* -» [9 -> 9']), unde 9 este 0 formulă din *L,* 9' este rezultatul înlocuirii unora dintre apariţiile libere ale lui *x* cu *y.*

Definiţia 1.8.14. *Procedeele de deducere ale calculului predicatelor* sînt :

i) *modus ponens* : formula este o consecinţă imediată a formulelor 9 şi 9 —> <p, oricare ar fi formulele 9 şi ţp •

ii) *generalizarea* : formula (Vx) 9 este o consecinţă imediată a formulei 9, unde 9 eslc o formulă şi x este o variabilă.

Definiţia 1.8.15. O *demonstraţie* a unei formule 9 este un şir fiind de formule 9J 9„ = 9 astfel incit pentru orice formulă 9, să avem una din urm ăl oarele trei situaţii :

i) 9(esle o axiomă.

ii) exislă *j. k < i* astfel incit 9$_{f}$ esle o consecinţă imediată a formulelor 9; şi 9J. (prin regula modus ponens).

iii) există *j < i* astfel incit *este* o consecinţă a formulei 9,- (prin regula de generalizare).

O *formulă* 9 este *inferabilă* dacă există o demonstraţie a ei. Acest lucru va fi notat prin 1-9, iar cu I)EM vom nota mulţimea formulelor inferabile ale calculului predicatelor.

Teorema 1.8.5. *Dacă* 9 *este o formulă a calculului predicatelor care se obţine dintr-o tautologie ct a calculului propoziţiilor prin înlocuirea variabilelor prin formule atunci* 9 s **DEM.**

Demonstraţie. Afirmaţia se obţine observînd că axiomele A.i) — vii) sînt de fapt axiomele calculului propoziţiilor in care

simbolii variabilelor an fost înlocuiţi cu formule, iar modus ponens este un procedeu de demonstrare comun calculului propoziţiilor şi calculului predicatelor. ■

Teorema 1.8.6. *Pentru orice formulă* 9 (x, y) *a calculului predicatelor avem*

I- (Vx) (Vy) 9 (x, *IJ)* (Vy) (Vx) <p(*x, y).*

Demonstraţie. în baza axiomei B i) avem :

I- (Vx) (Vy) 9 *(x, y)* - (Vy) 9 *(r, y)*

şi

I- (Vy) ? (*, y) - *? (x, y).*

Cum formula (a p) -» [(p -» y) -» (a -> v)J este o tautologie aplicînd teorema 1.8.5 găsim :

l_ [(Vx) (Vy) 9 *(x, y) -** (Vy) 9 *(x,* y)J -> [((Vy) 9 *(x, y)* -> 9 (x, y)) -

- ((Vx) (Vy) 9 (x, y) -» 9 (x, y))J.

în baza regulii modus ponens avem

I- ((Vy) 9 (x, y) -» 9 (x, y))((Vx) (Vy) ?(x , y) -> 9 (x, y)),

şi aplicînd încă o dată aceeaşi regulă obţinem

I—(Vx) (Vy) 9 (x, y) -> 9 (x, y).

Prin generalizare putem scrie :

I— (Vx) (Vy) 9 (x, y) -» 9 (x,y)

şi aslfel, prin axioma B ii) şi prin regula modus poncns :

H (V.r) (Vy) ? (x, y) (V.r) *z*> (,r. y)

în sfîrşil, folosind regula de generalizare, axioma B ii) şi modus ponens avem

I- (V.r) (Vy) (.r, y) *(Yy)* (V.r) ? (r, y). ■

Consecin(a 1.8.2. Ordinea in care se plasează cuantificatorii alunei cind se consideră închiderea formulei 9 (Xj ,r„) este indiferentă.

Obseri>a(ia 1.8.12. O formulă este inferabilă dacă şi numai dacă închiderea ei esle inferabilă. Acest fapt rezultă din axioma B i) şi din existenţa regulii de generalizare.

Obsernafia 1.8.13. în calculul propoziţiilor formulele aveau una din „valorile de adevăr" : 0 sau 1. în calculul predicatelor pentru a interpreta o formulă ca adevărată sau falsă trebuie să precizăm semnificaţia simbolilor predicativi precum şi valorile atribuite simbolilor variabilelor.

Definiţia 1.8.16. O *structură relaţională* esle un cuplu 21 = <A,{p„|ne A'}> unde *A,* domeniul lui 21 este o mulţime nevidă şi p_B este o relaţie pe A oricare ar fi *n* s iV.

Structura relaţională 21 este o *realizare* a limbajului calculului predicatelor dacă *a* (p_n) = y *(n),* unde *a* (p_B) este aritatea relaţiei p_B , Vn s N. Vom numi p„ *valoarea predicatului* P_n în realizarea 21.

Definiţia 1.8.17. O 21 - *valuare a variabilelor calculului predicatelor* esle un şir de clemente din *A :*

■T = (-^{I}o > -1'1 ,•••> x_n ,...)

Definiţia 1.8.18. Valuarea *x* satisface ? în 21, ceea ce se notează cu 211= 9, *X* dacă :

I 21V_m = V„ dacă şi numai dacă x_m esle acelaşi cu x_n.

ii) 5I|-^P_B ("<. ", $\overset{x}{\underset{o(n)}{}}$) dacă Şi numai dacă < ,$i_{(1}$...., r, >e p„.

iii) 21$\overset{x}{1}$=~ 1<p dacă şi numai dacă nu avem 211= ?

iv) 21$\overset{X}{1}$= p A dacă şi numai dacă 211= $\overset{X}{p}$ şi 211= $\overset{XX}{}$

v) 21$\overset{x}{1}$— (3/>$_B$)p dacă şi numai dacă există un *a* aslfel îneît 21 — p . unde .r„ (u) = (.1^, a-'»-!> n, .t_{B+1}

Vom scrie 211= 9 [a’$_o$ dacă orice valuare care atribuie simbolului
v_t valoarea x_t , *i n* satisface pe cp în 21. Formula 9 este *valabilă* în 21 dacă 21 |=<p oricare ar fi .r, ceea ce se va nota cu 21' = 9. Dacă 211= 9 oricare ar
X fi 21 atunci 9 este o formulă *universal valabilă.*

Se poate arăta :

Teorema 1.8.7. *Mulţimea formulelor universal valabile coincide cu* **DEM.**

Demonstraţie. Vezi [Beli, J. L. ; Slomson, A. B. 1, p. 63]. ■

PARTEA A II-A

LIMBAJE FORMALE

§ II. 0. Introducere

Limbajele formale constituie modele matematice attt ale limbajelor naturale, cit şi ale limbajelor artificiale, folosite pentru comunicarea cu maşinile electronice de calcul (limbaje de programare).

Se prezintă o ierarhizare a claselor de limbaje formale unanim cunoscută şi anume ierarhia lui Chomsky. Fiecare clasă a acestei ierarhii este studiată, attt din punct de vedere al generării limbajelor care fac parte din ea, cit şi din punct de vedere al recunoaşterii acestor limbaje cu ajutorul unor „maşini matematice". Sini cercetate d« asemenea proprietăţile de închidere ale claselor de limbaje faţă de diferite operaţii, precum şi raportul dintre diversele clase.

§ II. 1. GRAMATICI

Noţiunea matematică de gramatică este o formalizare a metodelor d« construire a limbajelor asupra unor alfabete.

Definiţia II.l.l. O *gramatică* este un 4-uplu G = *(I_N, I_T , x_0 ; F)* **unde** *I#* şi I_T sînt două mulţimi disjuncte, finite numite respectiv *alfabet neterminal* şi *alfabet terminal,* ,r_0 *e* J_N este *simbolul iniţial,* iar *F* este *mulţimea regulilor de generare* adică o mulţime finită de perechi ordonate (p, ţ)cupe (l^ţj U *1T P* conţinînd cel puţin un simbol din I_N , iar *q* G *(I_N* ţj *I_T)**

Dacă *(p, q)* s *F* vom nota acest lucru cu *p -> q.*

Definiţia II.1.2. *Cuvîntul p generează direct cuvîntul q în gramatica G,* dacă *p = r_1 p' r_2 , q = q' r_2 , p' ->■ q'* şi *, r_2* G /*,

unde I = Zj, tj I_T , ceea ce se va nota cu $p \Rightarrow q$.
o

Cuvîntul p generează cuvîntul q în gramatica G dacă există un şir finit de cuvinte $p = p_0, p_1 \quad p_n = q$, p_j s /*, j = O, n astfel ca $p_j \Rightarrow p_{j+i}$,

$G\ j =$ 0, n - 1.

Vom nota aceasta cu $p \overset{a}{\Rightarrow} q$.

Şirul p_0 , p, p_a formează o *deducţie* a cuvîntului p_n din cuvîntul p_0 în gramatica G, iar n este lungimea deducţiei.

Vom mai folosi şi notaţia :

$$Po \Rightarrow^* Pi \Rightarrow - \Rightarrow^* Pn-$$
$$a \quad o \quad o$$

Observaţia 11,1,1. Relaţia „=>“ este închiderea tranzitivă şi reflexivă a G relaţiei în cazul în care nu există pericol de confuzie vom nota, pentru
G
simplitate, relaţiile de mai sus prin => respectiv =>.

Definiţia II.1.3. *Limbajul generat de gramdtica G* este mulţimea :

$$L(G) = \{p \mid p\ V_T, x_0\ p\},$$

adică L (G) constă din mulţimea cuvintelor care sînt generate în gramatica G din simbolul iniţial x_0 .

Definiţia II.1.4. Două *gramatici* $G_{\pm}$ şi **G_2** sînt *echivalente* dacă L (GJ = L **(G_2).**

Exemplul II.1.1. Fie gramatica G_r = ({x_0}, {ij, i_2}. x_0 ; {x_0 -> ii i_2, x_0 -> ij x_0 i_2}). Atunci L (G_t) = {iV"$_2$ I n s N*}.

într-adevăr, $i_t\ i_2$ *e* L (GJ deoarece avem $x_0 \Rightarrow i_t\ i_2$. Apoi ij *iţ* = $i_{\pm}\ i_r\ i_2\ i_2$ e e L *{Gj)* din deducţia $x_0 \Rightarrow i_t\ x_0\ i_2 \Rightarrow i_t\ ii\ i_2\ i_2$. Avem *if* i_2 L (Gj) deoarece XQ ~'* ly XQ itj —*r* l'i li XQ l"$_2$ l_2 —*r'* ?i ?i îi l"$_2$ î$_2$ îij , Ş. 3. IU. d.

Exemplul II. 1.2. Gramatica G_2 = ({x_0}, I_T, $^x o$; {^{x}o "* *Pi* >•••• ^{T}o Pn})> unde py s *I* , j* = 1, n, generează limbajul finit L *(G_2) = {$p_r p_n$}.*

Exemplul 11.1.3. Gramatica $G_{(n)}$ = ({x_0}, {iji„ , $k_1 k_n$}, x_0 ; x_0 ->

e, x_0 x_0 ij x_0 k_Y x_0 , x_0 -» x_0 i_n x_0 $A_n x_0$}), *n e N*,* generează *limbajul n-paran- tetic* sau *n-limbajul de tip Dyck.*

Luînd $n = 1$ avem $G_{(1)}$ = ({x_0} ; {i_t, *kj,* x_0 ; {x_0 -» *e, x_0* -» x_0 ii *x_0* k_r x_0}). Iată cîteva cuvinte din *L* (GQ;) CU deducţiile lor :

$^x o\ ^x o$ **l"i *o**	**^'i** $^x o\ ^x o$ **h •Ťo** $^x o$ **l"l ^1 '**
X_o =* IQ il *XQ* *XQ* =>Ij	*ky XQ* => *XQ* ij XQAj *XQ* XQ AJ Ai ,
XQ =S> XQij XQ XQ =>îj	AJ XQ => XQ ij XQÎJ XQ AJ XQ Aj l\ A] A] .

în n-limbajele de tip Dyck în locul simbolurilor ij... i„ , k_4 ..... k_n se pot folosi paranteze de diferite tipuri (,..., (,)) de unde denumirea de limbaje
n-parantetice pentru aceste limbaje. $^{(1)}$ $^{(n)}$ $^{(1)}$ $^{(n)}$

De exemplu, pentru l-limbaje se folosesc în locul lui , A_x respectiv parantezele „(“ şi „)“. Corespunzător, cuvintele considerate mai sus se scriu (), () () Şi (())•

Exemplul II.1.4. Fie gramatica G_3 = ({x_0}, {P, 1,/\, x_0 ; {x_0-* -> T. 1 > XQ P 1, X_o -» A *Xg Xg* , XQ -» V xg *Xg* , XQ “> "| IQ}).

Limbajul *L (G3)* constă din formulele bine construite ale calculului pro- poziţional.

Astfel, primele două reguli dc generare ne permit construirea variabilelor „indexate'[1] P1, pil, pili,... Regulile de generare x_0 -> A x_0 x_0 şi x_0 -» V x_0 x_0 permit construirea conjuncţiei respectiv a disjuncţiei a două variabile, iar x_n -»1 x_0 construirea negaţiei unei variabile oarecare.

Exemplul II.1.5. Fie gramatica G_4 = ({x_0}, {i\ , i_2 i„ , V, *, 0, (,)}.
x_0 5 {*0 -* R >•••> $^x o$ '» > $^x o$ -> 0- x_0 -► x_0 V x_0 , x_0 -> (x_0 x_0), x„ -> a-;}). Limbajul *L (G4)* constă din expresiile regulate pe alfabetul {ij,..., i„} (v. I. Creangă, C. Reischer, D. Simovici [1], §11.14).

§ 11.2. IERARHIA GRAMATICILOR

Gramaticile pot fi clasificate impunind anumite restricţii regulilor de generare.

Definiţia II.2.1. *Gramaticile de tip O* sint gramaticile care nu au restricţii asupra regulilor de generare.

Gramaticile de tip 1 sînt gramaticile ale căror reguli de generare au forma :

$$\text{Pi } x\ p_2\ \text{P]}\ \text{pp}_2 \blacksquare$$

unde

$$\text{Pi} \bullet P_2\ ^{\text{s}}\ \text{IAU Ar)*- x e } I_N \text{ şi p e (ZJVU I}_T\text{)'\{e},}$$

sau

$$x \text{ -» } e,$$

şi x s I_y nu apare în partea dreaptă a vreunei reguli de generare.

Gramaticile de tip 2 sînt gramaticile ale căror reguli de generare sînt de forma :

$$x \text{ -* } p,$$

und

$$x \in (V_N \cup V_T)^*$$

Gramaticile de tip 3 sînt gramaticile ale căror reguli de generare sînt de

forma :

$$x \text{ -» } px, \text{ ,}$$

unde

$$x, x, e\ 1_N \text{ şi } p \in r_T$$

sau

$$x \text{ ^- } p,$$

unde x e , p <= /j.

Definiţia II.2.2. Un *limbaj de tip j* este un limbaj generat de o gramatică de tip *j, j* = 0,3

Vom nota cu $J>_j$ mulţimea limbajelor de tip *j.*

Observaţia 11.2.1. în cadrul gramaticilor de tip 1 un simbol neterminal *x* poate fi înlocuit printr-un cuvînt *p* dar numai în contextul p_x xp_2. De aceea aceste gramatici se mai numesc *gramatici „context sensitive" (sensibile la context),* iar limbajele generate de ele se numesc şi *limbaje „context sensilive".*

în gramaticile de tip 2 un simbol neterminal *x* poate fi înlocuit cu un cuvînt *p* independent de contextul în care apare *x.* De aceea aceste gramatici se mai numesc şi gramatici *„context free" (libere de context),* iar limbajele generate, *limbajele „context free".*

Obseivaţia 11.2.2. Are loc şirul de incluziuni stricte

$$£_0 \Rightarrow \quad =) J2_3,$$

aşa cum vom arăta ulterior (v. teorema II.6.1 şi exemplul II.6.1 şi teoremele II.10.8 şi II.15.3).

în continuare ne vom ocupa de unele proprietăţi de închidere ale limbajelor faţă de operaţiile de reuniune, produs şi închidere prin produs a limbajelor (v. §1.3).

Teorema II.2.1. *Familiile* Jl_j *(j* = 0, 3) *con(in toate limbajele finite şi sînt închise faţă de operaţia de reuniune.*

Demonstraţie. Limbajul vid este generat de orice gramatică pentru care I_T = 0. O astfel de gramatică este de orice tip. Limbajele finite sînt generate de gramatici similare aceleia

descrise în exemplul II.1.2, care este de asemenea o gramatică de orice tip.

Fie gramaticile $G_s = (I_{Xf}, I_{Tj}, x_{oj}$; F$_{()}$), j = 1, 2 de tipul k şi L (G_j) limbajele generate. Putem presupune l_{Ni} Q 1_{N2} = 0, deoarece în caz contrar vom atribui cîte o etichetă distinctă fiecărui simbol din cele două alfabete.

Limbajul L(G$_1$)ţJ L (G_2) este generat de gramatica:

GU = (Gl > ^*Tl* U ^*T2* ' *Ho* • U 1*2 U {Vo *01 ' *Ho* "* •r02}) '

unde

Dacă gramaticile şi G_2 sînt de tipul *k,* atunci şi gramatica Gj este de tipul *k.*

Avem *L(GJJ) —LţGJlJ L (G₂).* într-adevăr dacă p sL (Gj) atunci $y_0 \Rightarrow$ p. Or, .

aceasta presupune că în deducerea lui p a fost utilizată una din regulile de generare $y_0 \to x_{ol}$ sau $y_0 \to x_{02}$. în primul caz p e *L* (GJ, iar în al doilea caz p s *L* (GJ. Deci *L* (Gj) c L(GJ|J *L (GJ.* Apoi, dacă p s L(GJ(J

(J *L* (G_2) avem sau $x_{01} \Rightarrow$ p sau $x_{02} \Rightarrow$ p, de unde obţinem $y_0 \Rightarrow$ p, ţinînd o, *c,* seama de prezenţa regulilor de generare $y_0 \to x_{01}$ sau $y_0 \to x_{02}$, deci p e *L* (Gj), adică L(GJU L (GJ c *L* (Gj). Astfel familia este închisă la reuniune, $k = 0?3$. ■

Teorema II.2.2. *Familia limbajelor libere de context, J>₂ este închisă la operabile de produs şi închidere prin produs.*

Demonstrafie. Fie gramaticile de tip 2, Gy = *(IN₁, I_Tf, X₀} ; FJ), j* = 1,2 Limbajul *L* (GJ *L* (G_2) este generat de gramatica de tip 2 :

$$G = (J_{NI} \cup \check{A}_{N2} \cup \{l/o\}' \ \hat{}T1 \cup \hat{}T2 > 9o\ ' \ \hat{}1U\ F2 \cup \{ A/o\ *01\ *02\})'$$

unde Po^fjyiU -^2 • Dacă p e L,(G) atunci $y_0 \Rightarrow$ p, ceea ce revine la: *a* y_0 '=* *oi *02 =* *P*- Atunci p = pj p_2 , unde $x_{01} \Rightarrow p_x$, $x_{02} \Rightarrow p_2$, adică *P ˢ* Gj G_t s *L* (GJ *L (G₂).* Invers, dacă p = p_t p_2, unde $p_{\pm}$ s *L* (GJ, p_2 e *L (G₂)* avem $x_{01} \Rightarrow p_x$, $x_{02} \Rightarrow p_2$ şi deci $y_0 \Rightarrow x_{01}\ x_{02} =* p\hat{}\ p_2$, adică Pj p_2 e L (G). *a, o, a a*

Limbajul *(L* (GJ)* este generat de gramatica de tip 2 :

$$G. = (^{i}NI \cup \{l/o\} \to hi\ .\ J/o > F i \cup \{l/o\ ^{e} > l/o\ l/o\ *oil)«$$

unde y_0 $ I_{N1}.

într-adevăr, dacă p e (£, (GJ)* atunci p = $q_1 q_2 \ldots q_k$, unde q_t e *L* (GJ, decix$_{01}$ 4» q_t, *J* = 1, *k.*

Avem

$$l/o \Rightarrow l/o\ *oi \Rightarrow_{G_t} l/o\ 9i =*_{G,} l/o\ *oi\ 9* =* l/o\ ?t\text{-}i\ 9t =*$$

$$\bullet\bullet\bullet =*_{G.\,G.} l/o\ 9i\ 92 \qquad 91_{G.}\ 9i\ 92 \bullet\bullet\bullet 9t = P>$$

adică p e *L* (GJ.

Invers, dacă p e *L* (G)* trebuie să arătăm că p = $q_1\ q_2 \ldots q_t$, unde

x_{01} $=>q$ $_{t,}$ *a.*

$j = 1$, *t.* Avem $y_0 =t> p.$
G.

Plecînd de la simbolul iniţial y_g trebuie să aplicăm mai întîi, fie regula de generare y_g -» e fie regula de generare y_g -» $y_g\ x_{Oi}$.

în primul caz $p = e$ şi *e e (L* (GJ)*. în al doilea caz :

• *

Ho *Uo* *01 *P'*

Mai departe, în deducţia $y_0\ x_{gl} => p$ prima regulă de generare aplicată poate fi y_g -> *e,* în care caz *p* e *L* (GJ, sau y_g -» $y_g\ x_{Ol}$. Apoi *p* poate aparţine lui *(L* (Gj))^2, dacă în deducţia $y_g\ x_{01}\ x_{01}$ => *p* prima regulă utilizată a fost y_g -> *e* sau lui (L (GJ)*, $k > 2$, în caz contrar. ■

Teorema II.2.3. *Familia limbajelor de lipul* 3, $_JP_3$ *este închisă faţă de operaţiile de produs şi închidere prin produs.*

Demonstraţie. Fie *G, = (I*$_{Nj}$*, I*$_{Tj}$*, x*$_g$*) ; F*$_f$*),* $j = 1,2$ două gramatici de tip 3. Atunci gramatica de tip 3 :

G = (fjvi U *Ij/2 > I*$_{TI}$ U *I*$_{T2}$ *' ^"oi ' F* U *F*$_2$*),*

unde *F'* se obţine din $F_{\pm}$ prin înlocuirea regulilor de generare de forma x -» p, cu x s I_{N1}, *p <= I*$_{T1}$ prin regulile de generare *x* -» px_{oa}, generează limbajul LţGJLţG,).

• •

Dacă *p* <s *L* (Gj) *L (G*$_2$*)* atunci *p = p*$_Y$ *p*$_2$, unde $x_{01} => p_1$, $x_{g2} => p_2$

Atunci în gramatica G, x_{01} =s> $p_t\ x_{02}$ => pj p_2 = p, deci p *e L (G).* Invers, dacă p s *L (G)* avem x_{01} => p, adică $x_{01} => q_t => q_2 => ... => q_{„} = p.$ Fie q_t primul cuvînt de forma $p_2\ x_{02}$. Atunci pj s *L* (GJ şi pi x_{02} => pj p_2 = p, unde p_2 G *L* (G_2).

Apoi, gramatica de tip 3

G» = *(Im* U {</<>}> > *Do > F*| U *F* U {j/o -* ^oi > ^{r}oi ^{e}})'

unde $y_0\ I_{N1}$ şi *F"* se obţine din F_2 înlocuind regulile de generare, de forma x -» p, *x* s I_{Nl}, p e F_{T1} prin regulile de generare de forma x px_{Q}} , generează limbajul *(L* (G_x))*.

Dacă p e (L (GJ)* atunci p = $q_l\ q_2 ...\ q_t$, unde s *L* (GJ, deci x_{01} => q_{jt}

——

$j = 1, k.$

Apoi $y_0 \Rightarrow x_{01} =^* q_1 x_{gl} =s> q_1 q_2 x_{01} \Rightarrow \dots \Rightarrow q\hat{} q_2 \dots x_{01} =» q_x q_2 \dots q^*$, aplicînd regula de generare $x_{01} \to e$, deci $(L(G_x))^* \pounds L(G^*)$.

Să arătăm incluziunea inversă, $L(G,) \pounds (L(G_x))^*$.

Fie p «s L (G»). Trebuie să arătăm că p = $q_x q_2 \dots q_t$, unde oi ' o. **1,** t sau p = e.

Relativ la mulţimea F_p a regulilor de generare aplicate pentru obţinerea cuvîntului p distingem următoarele cazuri :

a) $F_p = \{y_0 \to x_{01}; x_{01} \to e\}$; atunci $p = e$ şi p e (L (GJ)*,

b) $F_p = F_{lp}$ u $\{y_0 \to *oi\}>$ unde $F_{lp} \pounds F_x$; atunci p e $L(G_x) \pounds (L(GJ)^*$.

c) $F_p = F_p$ u $\{y_0 \to *01 > *01 -* e\}>$ unde F p ^F" > atunci

J/o "* ^oi 7i oi 71 72 *oi ••• 71 72 ••• 7t * oi 7i 72 ••• 7e e (L $(G_x))' \pounds (L(G_x))^*$.

d) $F_p = F_{lp}$ u U $\{p_0 \to *oi, *oi - e\}$, unde $F_{lp} \pounds F_x$, F" £ F" ; atunci $y_0 -* x_{01}$ i p şi p e $L(G) \pounds (L(G_x))^*$ sau $y_0 -* r_{01} \Rightarrow q_1 z_{01} p_x p_2$ e

$^eL(Gj)^2$ sau y_0-* *oi => ?i *oi 7i ?2 *oi Ş.a.m.d., adică la fiecare pas putem aplica o regulă de deducţie din F_{lp} sau din F_p sau rr_{01}-»e. În toate situaţiile obţinem un cuvînt p e $(L \pounds (L(Gj))^*$.

Cazurile a) — d) epuizează toate posibilităţile, cum se constată imediat. ■

Teorema II.2.4. *Familia de limbaje $J>_0$ este închisă la operaţia de produs.*

Demonstraţie. Se procedează la fel ca în cazul teoremei II.2.2 ■

Teorema II.2.5. *Familia de limbaje este închisă la operaţia de produs.*

Demonstraţie. Fie gramaticile G, = I_{Tj}, r_{Q1}; $F\hat{}$, $j = 1, 2$ două gramatici de tip 1.

Să considerăm gramatica :

G = (fjviU *Ift2* U {Po}> ^riU *ITZ* > 7o > F),

unde y_0 $ I_{N1} u/$_{W2}$, $I_{Nl}r$> I_{N2} =0 (restricţie care nu micşorează generalitatea), iar pentru F distingem următoarele cazuri :

a) F = FJUF$_2$U {y_0 -> x_{01} *$_{02}$} dacă (r_{01} -> e) F_x şi (r_{02} -> e) $= F_2$ deoarece într-o gramatică sensibilă la context putem avea reguli de generare de forma *x -» e* numai pentru simboli care nu apar în partea dreaptă a unei reguli de generare or, în gramatica G avem regula de generare y_0 -» *$_{01}$*$_{02}$.

b) F = (F_2\{T_{01} - e}) u F_2 U {p_0 - *oi *02 5 J/o *02}. dacă (x_{c2} -* e) $ F_2 .

c) F = F_x u (F_2\{*$_{02}$ -> e}) u {p_0 -> *$_{01}$ *$_{02}$ J Po "* ^{x}oi}> ^{dacă} (^{x}oi ■* ^{e}) £ ^{F}*i* •

d) F = (F_x \ {x_{01} -» e}) u (F_2 \ {*$_{02}$ -» e}) u {p_0 ■* *01 *02 ! Po ^{e}}-

Gramatica G este de tipul 1, cum se constată imediat ţinînd seama de definiţia II.2.1.

Avem *L* (G) = *L* (GJ *L* (G_2), ceea ce se obţine printr-un raţionament identic cu cel folosit în teorema 11.2.2. ■

Teorema II.2.6. *Familiile de limbaje J>$_a$ şi JPJ sini închise la operaţia de închidere prin produs.*

Demonstraţie. Fie gramatica *G = (I_N, I_T, x_0 ; F)* de tipul *j, j* = 0, 1.

Limbajul (L (G))* este generat de gramatica de tipul G. = *(I_N* u *{y_0 , y_L}, 1*, V_Q ; F,),* unde *F, = F_x* u {t/$_0$ -> *e, y_0-> x_0, y_t -> x_0 y^}* u *[iy* -► *ix$_0$, iy$_x$* -» -► iz,, p$_{1(}$ Vi e *I_T},* unde :

$$F_1 = \begin{cases} F\text{'}\ daCă\ J= \ o \\ \text{F\{r}_{0}\text{-}_{>}\ e\}, \text{ dacă } j = 1. \end{cases}$$

Avem (L (G))* £ *L (G_t).* într-adevăr, fie *p = p_l p_2... p_t* e (L (G))* £ S *(L* (G))*, adică x_{Q_1} => p_f *j* = 1, *k.* în gramatica $G_\#$ avem deducţia : *ó*

Ho -* ^{x}o Pi *Pi Hi Pi* •^{T}o Pi *Pi P2 Ui =* Pi P2* ■■■
o o. o.

••• *Pk-l 111* ••• => *Pl P2* ■■■ *Pk—1* ^{X}*0* Pl *Pi* ••• *Pk-l Pk-*
a, a, a.

Invers, *L* (G,) £ *(L* (G))*. Astfel, fie *p <= L (G_t).* Prima regulă de generare aplicabilă în gramatica G« poate fi :

1) y_0 -*■ *e* ; în acest caz *p = e* şi *p*(L (G))*

2) y_Q -» x_0 , de aici rezultă p e L (G) £ (L (G))*

3) J/o “* ^{x}o !/i ’ dacă în gramatica G nu există deducţia z_0 => p. In această

situaţie $p = q$ unde $x_0 \Rightarrow p_{p_2 y_t}$, ^{a}. In continuare i/$_0$ -> $\overset{a}{a^*}$ .r$_0$ $y_{Y_i} p_1$ $y^{\wedge}$ p_x G*

dacă $p=p_1p_2q'$ sau y_0 x_0 y_Y => p_l y_t p_T $x_{0\,Pl}$ p_2 dacă $p = p_t$ p_2 *c.* c. ş. a. m. d.

Procedeul se termină după un număr finit de paşi, deoarece lungimea lui p este finită şi obţinem p s (L (G))*. ■

Teorema II.2.7. *Familiile de limbaje , j* = 0,3 *sini închise la operaţiile de reuniune, produs şi închidere prin produs.*

Demonstraţie. Rezultă din teoremele II.2.1 — II.2.6. ■

Observaţia II.2.3. Dacă, limbajul L este de tipul j atunci limbajele L u {e} şi L | {e} sînt de acelaşi tip *j, (j* = 0,3).

§ II. 3. AUTOMATE FINITE ŞI LIMBAJE REGULATE

în continuare ne vom ocupa de dispozitivele care „recunosc*' limbajele **generate** de gramaticile de tipul 3, adică de automatele finite.

Definiţia II.3.1. Un *automat finit* este un 5-uplu $A = [/, S, O, f,$ <7] unde *I, S, O* sînt mulţimi finite nevide numite respectiv *alfabet de intrare, mulţime de stări* şi *alfabet de ieşire, f:Sx I* <p*(S), *g* : S x *I -> (Ţ>* (O)* sînt *funcţiile de trecere şi de ieşire* ale automatului.

Dacă Vs e S, i s /, | *f* ($, j) | = | *g (s, i)* | = 1, atunci A se numeşte *automat determinist.* în caz contrar *A* este un *automat nedeterminist.*

Observafia 11.3.1. în toate cele ce urmează consideraţiile referitoare la noţiunea de automat sînt valabile atît pentru automatele deterministe cît şi pentru cele nedeterministe. Proprietăţile specifice unuia din tipurile de mai sus vor fi precizate ca atare. Pentru simplitatea expunerii un automat' finit va fi numit pe scurt, automat.

Observaţia 11.3.2. Noţiunea de automat finit intră în sfera mai largă a noţiunii de sistem cibernetic, care este un model al dispozitivelor de prelucrare a informaţiei provenind din cele mai diverse surse* funcţionînd la intervale de timp bine determinate numite *tacte.*

Ne putem imagina un automat ca o cutie neagră prevăzută cu o bornă de intrare şi o bornă de ieşire, capabilă să posede un număr finit de stări' interne (elementele mulţimii S).

Aplicînd un semnal de intrare *i* (din alfabetul *1)* la borna de intrare sistemul respectiv elaborează un răspuns (un element din alfabetul *O)* şi trece într-o nouă stare internă. Acest răspuns precum şi starea în care trece automatul în tactul următor aparţin respectiv mulţimilor *g (s,* i) şi *f (s, i), s* fiind starea în tactul actual.

Există automate la care se impune ca starea în tactul iniţial al funcţionării lor să aparţină unei anumite mulţimi.

Definiţia II.3.2. Un *automat slab iniţial* este un 6-uplu *A* = [/, S, *O, f, g ;* S_o] unde *1, S, O, f, g* au semnificaţiile din definiţia II.3.1, S_o s S şi starea automatului în tactul iniţial aparţine lui S_o . Dacă | S_o | = 1, *A este* un *automat iniţial** iar dacă S_o = *S, A* este un *automat neinifial.*

Definiţia II.3.3. Un *semiautomat* este un triplet *A* = [7, S, *f]* unde *1, S, f* au semnificaţiile cunoscute. Noţiunea de *semiautomat slab iniţial* se introduce ca în definiţia precedentă.

Definiţia II.3.4. *Semiautomatul ataşat* automatului *A* = [7, *S, O, f,* <7] este semiautomatul S (A) = [/, S, /"].

Observaţia 11.3.3. Prin considerarea semiautomatului S (A) se face abstracţie de comportarea la ieşire a automatului A rămînînd esenţială funcţionarea internă a automatului.

Remarcăm că unui automat i se poate ataşa în mod unic un semiautomat, iar un semiautomat poate fi extins la un automat prin alegerea unui alfabet de ieşire *O,* şi a unei funcţii de ieşire *g.*

Automatul astfel ataşat nu este unic el depinzînd de această alegere.

Introducerea în *n* tacte succesive a cîte unui simbol i_x, i_2,..., i_n revine în fond la introducerea cuvîntului *p* = i, ... i_n s *I**. Apare necesară astfel

extinderea funcţiilor extindere se va face în de trecere şi de ieşire de la S x 7 la S x 7*. Această mod inductiv asupra lungimii cuvintelor din 7 şi anume :

(II.3.1) $f(s, e) = 5,$

(II.3.2) $f(s,pi) = f(f(s, p),$ i),

(II.3.3) $9(s, pi) = g(f(s, p), i).$

Observaţia II.3.4. Se constată imediat că pentru $p = e$ din formulele

(II.3.4) /■(«. ei) = f(f(s, e), i) = f (s, i),
9 (ˢ> ei) = g (f (s, e), i) = g (s, i),

deci funcţiile extinse coincid pe S x 7 cu funcţiile automatului *A.*

Teorema II.3.1. *Fie A =* [7, S, *0, f,* j]. *Pentru* Vs *e S, i <s I, p, r e* 7* *au loc :*

(II.3.5) $f(s,ip) = f(f(s,i), p)$

(II.3.6) $f(s, P^{\wedge} = f(f(s, P), r)$

(II.3.7) $9(s, ip) = 9(f(s, i), p)$

(II.3.8) $9(«. pr) = 9(f(s, p), r).$

Demonstraţie. Vom demonstra formula (II.3.6) prin inducţie asupra lungimii cuvîntului *r,* demonstrarea celorlalte relaţii făcîndu-se analog.

Pentru *l* (r) =0, r *= e,* formula (II.3.6) este valabilă ţinînd seama de formula (II.3.1). Să presupunem formula adevărată pentru *l* (r) = *n* şi să o demonstrăm pentru *q = ri, l (q) = n* + 1.

Conform formulei (II.3.2) şi ipotezei inductive se obţine :

/■(*. P<7) = /■(*> *pri) =f(f & pr), i) =*

= /■(/■(/■ (*> P). O. 0 = *f(f (s, p), ri) = (f (f* (s, *p), q).* ■

Vom mai folosi funcţiile f_s *: I* -» <Ţ>* (S),* f_9*: S -» <Ţ>* (S), g, t I* -> <p** (O), : 3 -» *(O)* definite respectiv prin :

f, (P) = f_p (ˢ) = *f* (ˢ> P)>

9, (P) = *9p («) = 9* («. P)-

Pentru Sj £ S şi *E* s *I** vom mai folosi notaţia :

$$f(Si, E) = \bigcup_{\substack{teSi \\ pGE}} f(s, p), \quad 9(S_{lt}E) = \{J\, g(s, p). \quad p\&E$$

Fie *A = [I, S, O, f, </]* un automat. Oricărei stări s s S îi

corespunde o aplicaţie între semigrupurile libere *I** şi O* aşa cum rezultă din :

Definiţia II.3.5. *Operatorul ataşat stării s* este aplicaţia *: I* -» O** definită prin

Y« Oi '2 •••'») — *9>(ii) 9/(t,ti)* O2) ('»)' Vil $i_2... i_n$ S /*.

Exemplul II.3.1. Automatul $A =$ >}, {s_x, s_2, s_3}, {o_r, o_2}, *f*, *f* şi *g* sînt definite prin următoarele

f tSl S2 S3	*9*	Sj s_2 s_3
ii *S2* S1 S3	Zl	0 j *0* 2 01
<2 s_3 «2 ,S2	f2	O2 Oj Og

este un automat determinist. Avem, de pildă *(i_j i_2 i_Y i_2) =* *9/(>,,h)* O2) «1 <>) Oi) *9/(>i, n* <i<i) O2) = °i °i °i °2 •

Exemplul II.3.2. Automatul *A* =^2}' fcl - ^2}' {°1 > °2}> *Unde*

f S1 *S2 S3*	*9*	S1 S2 S3
<1 {^1 , $2} {si} {^2}	'1	{o/ {o_2} {OpOj}
'2 {^2} {ss} {Sl}	'2	{O1,O_2} {o_2} {01}

este nedeterminist. Aici y_{Ji} *(i_1 i_2 $i_\pm$ i_2)* = {oj {o_x, o_2} {(^ , o_2} {oj, *o_2}.*

Ne ocupăm în continuare de limbajele care pot fi recunoscute de automatele finite *deterministe.*

Definiţia II.3.6. *Limbajul reprezentat în automatul iniţial A* ^{f} **= [Z, S, O,** *fi 9* î sol Pnn *semnalul de ieşire o e O* este mulţimea *E* = <77/(0) £ *I*,* iar orice *p* e *E* va fi numit *cuvînt recunoscut de automatul A prin semnalul de ieşire o.*

Definiţia II.3.7. *Limbajul E este reprezentat in automatul iniţial A* = [I, S, *O, f,* <7;s_0] *prin mulţimea de stări S_Y* £ S *dacă E = f~x* (SJ.

Observaţia II.3.5. Definiţia II.3.7 se poate formula şi în cadrul mai res- Lrîns al semiautomatelor iniţiale deoarece ea nu face apel la comportarea externă a automatului.

Legătura dintre definiţiile precedente este dată în :

Teorema II.3.2. *Fie E = gŢj* (o). *Atunci pentru mulţimea de stări S_x = = f (s_a , E) avem*

E u {e}, *dacă s_0* e
E, tn caz contrar.

Demonstraţie. Dacă *p* e *E, f* (s_0 , p) e S; deci *p* e /ŢjţSJ adică *E*

£ /7}(Si). Invers, fie *p* e *f^ţSj).* Atunci $f_{la}(p)$ « Sj şi dacă p / e urmează că p e *E.* Dacă s_0 e S, atunci $f_{,t}$ *(E)* = s_0 s Sj şi în acest caz p e *E* u {e}. ■

Observata II.3.6. Teorema anterioară arată că reprezentarea limbajelor prin mulţimi de stări este mai generală decît reprezentarea lor prin semnale de ieşire deoarece în al doilea caz este exclus cuvîntul nul *e.*

în continuare, utilizînd numai reprezentarea limbajelor prin mulţimi de stări ne vom limita la semiautomate (v. observaţia II.3.5).

Limbajele reprezentabile în semiautomate finite vor fi numite *limbaje regulate.* Mulţimea limbajelor regulate din semigrupul liber *I* va fi notată cu *<£(I).*

Pentru automatele nedeterministe definiţia II.3.7 se extinde în mod natural.

Definiţia II.3.8. Un *limbaj E* s *I* este reprezentat în semiautomatul nedeterminist slab iniţial* >1 = [7, *S, f;* S_o] prin mulţimea de stări S, dacă

$$E = \{p \mid f(S_0, p) \cap Sj / 0\}.$$

Teorema II.3.3. *Mulţimea limbajelor regulate coincide cu mulţimea limbajelor reprezentate în semiautomate nedeterministe slab iniţiale.*

Demonstraţie. Cum orice semiautomat iniţial determinist poate fi interpretat ca un semiautomat nedeterminist slab iniţial rezultă că orice limbaj regulat este reprezentabil în semiautomate nedeterministe.

Reciproc, fie *E* un limbaj reprezentat în semiautomatul nedeterminist slab iniţial *A = [I, S, f;* S_o] prin mulţimea de stări Sj. Să considerăm semiautomatul determinist iniţial *A'* = [7, *<p*(S), F* ; S_o] unde

$$F(S',i) = u$$

• es-

Limbajul *E* coincide cu limbajul regulaL E^1 reprezentat în semiautomatul *A'* prin mulţimea de stări Z = |S'| S' A Sj / 0} £ *<Ţ>*(S).*

într-adevăr, fie p e *E.* Atunci *f* (S_o, p) fi Sj / 0 şi deci *F* (S_o , p) = = *f* (S_o, p) s Z, adică p e *E'.* Dacă p E *E', F* $(S_o$, p) E Z, adică *F* $(S_o$, p) A A ■$! / 0 Şi cum *f* $(S_Q$, p) = *F* (S_o, p) urmează p s *E.* ■

*Observafia II.3.*7. Utilitatea introducerii limbajelor reprezentate în semiautomate nedeterministe va reieşi din paragraful următor unde le vom folosi în demonstrarea unor proprietăţi de închidere pentru mulţimea <2 (7).

§ II. 4. LIMBAJE DE TIPUL 3

Teorema II.4.1. *Limbajele de tipul* 3 *coincid cu limbajele regulate.*

Demonstraţie. Fie $G = (7_K, I_T, x_0 ; F)$ o gramatică de tip 3 unde I_N — {xo > ,..., x„}. Regulile de generare în gramatica G pot fi fie de tipul $Xj \to p_t\ x_k$, fie de tipul $x, \to p_r$, unde $x_j, x_h \in I_N, p_t \in 7^*_T$. Să considerăm semiautomatul finit nedeterminist $A = [7_X, I_N$ u {x, g}, f; $x_0]$, unde 7j = {e} u U *{Pt* I p_t ^{s} *1*T,* 3.^ , *e* I_N, $(x_j - p_t$.$i_A)$ Fsau $(x_j \to p_k)$ e *F}* cu alte cuvinte alfabetul de intrare fj constă din mulţimea finită a cuvintelor care apar în partea dreaptă a regulilor de generare din Fia care se adaugă cuvîntul vid ; x, y £ I_N.

Notînd S_{lk} = [TJ / $(x_j \to p_k$ z$_M$) s F} definim f prin

$$\text{U s''' d''ă}$$

$$(y, \text{ dacă } (x_t \to p_k) \text{ £ F şi } S_M = 0$$

$$/\blacksquare(\bullet_T, p_t) = f\,(g, Pt) = y < *pt^{\,e}\ h\ \blacksquare$$

$$f(x_0,e)=x_Q.\ f\,(x_t\ ,\ e) = y,\ f\ (y,\ e) = y.$$

Vom arăta că limbajul L (G) coincide cu limbajul E, reprezentat în semiautomatul A prin mulţimea de stări Sj . unde :

$$_s\ ) \{*0 > *\}\bullet^{\text{dacă}} (^2\text{o } e)\ S\ F$$
$$\text{l } \{r\}, \text{ în caz contrar.}$$

Putem presupune, fără a restrînge generalitatea că dacă (.$r_0 \to e)\ e\ F$, atunci T_0 nu apare în partea dreaptă a vreunei reguli de generare din F, deoarece, dacă avem simultan (.r_0 -> e), $(x_j \to px_0)$ s F, putem înlocui regula de generare $(x_t \to px_0)$ prin $(Xj \to$ p), fără a schimba tipul gramaticii şi limbajul generat de ea.

Fie p = Pi $p_2 ... p_n\ ^e\ I$- (6), n 1. Atunci avem deducţia $x_0 \Rightarrow p_t\ x_h \Rightarrow — \Rightarrow Pi\ P2\ Pn\text{-}i\ ^x i_n P2\ Pn\ ^{\prime n}$ semiautomatul A obţinem şirul de stări : x_{it} e $f\,(x_0$. pj, x_h s $f\,(_{X/l}, p_2), ..., x$ e $f\,(x_{jn\ l}$, p„t, adică $xs\ f\,(x_0\ , p)$. sau p e E. Dacă e s /, (G), atunci ($x_0 \to$ e) s F, .T_0 s SJ şi e e E, de unde $L\ (G)\ s\ E$. Să arătăm acum că E s L (G). Fie p e E cu p e. Există atunci un şir de stări $i_0, x_{Jlt}\ x_h$ rr$_{>nl}$, x, astfel ca $x_h e\ f$ (_r$_0$, $_{Pl}$), x_{jt} s f (.$_{Tlj}$, p$_2$),...
...$_{XJ}$ e f (rtj* $_2$, Pn_i), $^{x<}= f\ (^x j_n$ j, $Pn)'$ deci există în F regulile de generare « $^x o \hat{}\ Pi^x ii\text{-}\ ^x i_l \hat{}\ P2^x i.\text{--}^x in\text{-}|\text{-}\hat{}\ Pn'$ $^{\text{adica}}$ există deducţia .$r_0 \Rightarrow p_t\ p_2$ ... p„. Q deci p = L *(fi).* Dacă p = e e E, atunci f (.r_0 . e) = $x_0\ S_t$. de unde ($i_0 \to e$) e F şi e s L *(G).*

Reciproc, fie E limbajul reprezentat în semiautomatul determinist A = = [Z, S, prin mulţimea de stări S; . Definim gramatica de tip 3, G =
= (S. I, s_0 ; F), unde

F = {(s> -> p_k s_h) / s_h = [(s, , pj.)} u {(«, *Pt)|f (•'•/, Pt)* ^{s} Sj}.

Să arătăm că *E = L (fi).* Fie p = i_k i_2... i_n s E, p # *e.* Există şirul de stări s_0 , Sj ,..., s_{fl} , unde s_r = *f* (s_0 , *i,),* s_2 = *f* (s, , i_2), ..., s_n= *f* (s_n^$_1$, i„), s_n s . Corespunzător, în gramatica G apar regulile de generare s_u -+ i_k , Sj -> i_2s_2,...

..., s_0 j ->■ i„ şi astfel există deducţia s_0 => i_1 i_2 .*.. i„ = p.* Prin urmare, p s *a* e *L (fi).* Dacă p = e, atunci s_0 e Sj şi avem regula de generare s_0 -* e, adică e G *L (G).* Astfel *E G L* (G). Invers, fie acum *p* s *L* (G), *p e.* Cuvîntul *p* se obţine atunci aplicînd succesiv regulile de generare de forma :

^{5}o"*'i^{s}i .... i„, ceea ce antrenează pentru automatul .4

existenţa şirului de stări s_0 $s_{„_1}$, s_n cu ,$s_{„}$ = *f*($s_{„_x}$, $i_{„}$ ^$_}$) e S, şi =

= - «»). A = 1. n - 1.

Atunci $s_{„}$ = *f* (s_0 . i_t i_2... $i_{n_}$,)e Sj şi *p* s *E.* Dacă *p = e.* atunci *p* a fost obţinut aplicînd regula de generare .s_n -» e, de unde *f* (s_0 , *e) = s_0 e S,* şi deci *e* G *E.* g

Exemplul II.4.1. Să considerăm gramatica de tip 3, G = (lx_0 , Xj}, {ij , i_2}. x_0 ; *F),* unde *F* = {x_0 -> i] Xj , Xj -»z\ x_t , Xj -» i_2 x_0 , x_x -»z'j}. In semiautomatul finit determinist *A* -. (x_0 , x, , x. z/}, *f;* x_0], unde I_j — {e, i, , i_2}, iar

f este dat de tabloul :

f	*x0*		*x*	*y*
e	{*«}	*{y}*	{</}	(.'/}
'1	w	{X, X,}	{.</}	*{y}*
'2	*{y}*	{•^{T}o}	1'7}	*{y}*

limbajul *L* (G) este reprezentat de mulţimea de stări {x}. Avem, de exemplu, *f*(r_0 , z'j z'i) n {x} / 0, deci z\ *i, L (G).* Apoi $i_ii_2i_ii_l$ e *L (G),* etc.

Teorema II.4.2. *Mulţimea limbajelor regulate este închisă la operaţiile de reuniune, produs şi închidere prin produs.*

Demonstraţie. Se ţine seama de teoremele II.4.1, II.2.1, II.2.3. ■

Teorema II.4.3. *Fie E* c *I*. Următoarele afirmaţii sini echivalente :*

1) *E este un limbaj regulai, E (J> (I)*

2) *E este o reuniune de clase de echivalenţă ale unei congruenţe de rang finit pe I*.*

3) *congruenta este de rang finit, unde*

P-^{E} = {(p. 9) I XB <^{r}*i P 'i) = 7.E* (>| $^(P_2Y$ ^{Vr}i ' ^{r}-> $^{e\ l}$*}-

Demonstraţie. Vom arăta că au loc următoarele implicaţii : 1) => 2), 2) =* 3), 3) ~ 1).

1) => 2). Fie *E* G *<T2(I)* şi *A* = [/, S, *f;* s_0] un semiauLomat iniţial, finit, în care *E* este reprezentat prin mulţimea de stări , adică p = E —• *f (s₀ , p)* e Sj.

Definim relaţia *ţ* = Q ker *f*, . *ţ* este o congruenţă pe *I*. ţ* este o echiva- • *es*

lenţă, fiind intersecţia unei mulţimi de relaţii de echivalenţă. Fie acum *(p, q)* G£. Să verificăm că *(pr, qr)* s *ţ, (rp,rq)* G £, Vr s *1*.* Avem *f, (rp) = (f (f (s, r),*

P)= f(f(s, r), q) = f,(rq), Vs ; lafel *f, (pr) = f (f (s, p),* r)= *f (f (s, q), r)* =

= f, (qr), Vs, adică *ţ* este o congruenţă.

Relaţia *ţ* este de rang finit.Fie S = {sj ,..., s_t} şi o aplicaţie *T : S ->■ S.*

Notăm *I* (T)* = {p | *p* s /*, *f (sh, p)* = - (s*); *h* = 1, *k}.*

Să arătăm că (p, *q)* s ţ dacă şi numai dacă există o aplicaţie r astfel ca {p, 7} c 7* (T).

Fie (p, *q)* <2 £. Definim aplicaţia r prin T (s_{ft}) = , p) = *f* (,s_{ft}, *q).* Avem

p 6 *I** (T) şi *q* G *I** (T). Invers, dacă {p, *q}* s *I** (r) atunci *f (sk, p)* = T (s_{fc}) = = *f (ˢh> <î)'* deci (p, *q)* e Astfel, fiecărei clase din mulţimea *I*/ţ* i se ataşează în mod biunivoc o aplicaţie r : S -» S. Deoarece numărul acestor aplicaţii nu poate depăşi Ă*, urmează | *I* / ţ* I < k^k , adică *ţ* este o congruenţă de rang finit.

Fie acum p G *E* şi (p, *q)* G £. Atunci *f (s₀ , q) = f (s₀* , p) G Sj , deci *q* e *E.* Astfel *E* este reuniunea unor clase de echivalenţă ale lui £.

2) => 3). Fie *ţ* o congruenţă de rang finit pe 7* şi *E* o reuniune de clase de echivalenţă după relaţia £. Fie (p, *q) e* Dacă y_E (r_i pr_2) = 1, atunci, deoarece *(j|pr₂, i|qr₂)* G £, obţinem (q qr_2) = 1, deci (p, *q)* G pA Prin urmare, *ţ* £ pA de unde | *I* / y.ᴮ* | < | 7* / *ţ* /.

3) => 1). Fie *E* conţinut în *I** şi să presupunem că | /* / jA | < 00. Limbajul *E* este reprezentat în semiautomatul iniţial *A* = [Z, 7* / *p.ᴮ, f;* [ej], unde *f* ([p], i) = [p i], prin mulţimea de stări S_t = {[p] | p G *E}.* Se observă că *f* ([p], 9) = [p?].

Avem p G *E* <— [p] G S_x — *f* ([ej, p) *G Sₜ.* ■

Teorema II.4.4. Fie *E £ I*. Următoarele afirmafii sînt echivalente :*

1) *E este un limbaj regulat, E* G rg (7)

2) *E esle o reuniune de clase după o congruentă la dreapta pe I*, de rang finit.*

3) *Congruenta la dreapta vᴮ este de rang finit, unde*

$$= \{(P> 9) > 7.E\ (P^r) = I_E\ (<ir),\ Vr\ G\ 7^*\}.$$

Demonstrafie. Se procedează analog, ca la teorema II.4.3,

definind încă, ţ = ker f_s,; se constată imediat că *ţ* este o congruenţă la dreapta. ■

Teorema II.4.5. *Dacă E (Q* (7), *atunci numărul minim de stări ale unui semiautomat în care E este reprezentabil este egal cu* | 7* / A |.

Demonstraţie. Procedăm prin reducere la absurd. Să presupunem că limbajul *E* poate fi reprezentat în semiautomatul *A'* = [7, S', *f*; s,] şi | *S'* / < < | 7* / A |.

Să considerăm relaţia £' = ker Ca in demonstrarea implicaţiei 2) => 3) din teorema II.4.3 se arată că | 7* / A | < | 7* / £' |.

Aplicaţia T : 7* / definită prin T ([p]) = *f','* (p) este injeclivă, deci

7* / £'| | S' |, ceea ce contrazice presupunerea iniţială. ■

Teorema 11.4.6. *E* G (7) *dacă şi numai dacă* E^c G (7), *unde* E^c = = 7* \ *E, adică* 62 (7) *este închis la complementariere.*

Demonstrafie. Dacă *E => f"*[1] (Sj), atunci E^c = /7,[1](S\ SJ, unde *A* = = [7, S, *f*; s_0] este semiautomatul în care *E* este reprezentat prin mulţimea de stări . ■

Teorema II.4.7. 62 (7) *este închis la intersecţie.*

Demonstraţie. Fie E\ , E_2 e (Z) şi = [Z, S^1, *p* ; .s£j, *j* = 1,2 semiautomatele în care limbajele E_t şi E_2 sînt reprezentate respectiv prin mulţimile de stări Sj c S^1, S_2 £ S^2. Să considerăm semiautomatul 4 = [Z, S^1 x x S^2, *f*; (sj , .<o)J. unde *f: S^1 x S^2 x I -*>■ S^1 *x* S^2, astfel ca pr_y *[= f, ,j* = 1, 2.

Limbajul E, 0 E_2 este reprezentat în semiautomatul *A* prin mulţimea de stări x S_2 £ S^1 x S^2. într-adevăr, dacă *p* e E_r n E_2, atunci f^1 (sj, p) e Sj , *p (,<*[2] *, p)* e S_2. deci *f* ((sj , <[2]). *p) e S, x* S_2 şi reciproc, prin urmare, E_t n E_2 *e (Q* (Z). ■

Observata II.4.1. Ţinînd seama de teoremele II.2.1, 11.4.1, II.4.6, se obţine imediat rezultatul precedent, deoarece E_t n E_2 — (Eţ u E_2)[c].

Teorema II.4.8. *Mulţimea* 62 *(I) este algebră Boole fală de operaţiile de reuniune, intersecţie şi complementariere de mulţimi.*

Demonstraţie. Rezultă imediat din teoremele precedente. ■

Putem regăsi un rezultat cuprins în teorema 11.2.1 pentru cazul *j* = 3, utilizînd exclusiv semiautomatele finite :

Teorema II.4.9. *Dacă* ; E j < + co. *atunci E* <s 62 (Z).

Demonstraţie. Este suficient să arătăm că dacă IEI = 1. adică E = {p}, atunci Zi s 62 (Z) Şi să ţinem seama de teorema II.4.8. ..

Dacă p = *e.* atunci. *E* poate fi reprezentat într-un semiautomat iniţial oarecare A = [7, S, *f*; .s_0] prin starea s_0 , *f* fiind ales ast fel. încît *f (s. i) s_0 ,* Vi e Z, A V.s e S.

Fie p = $i_1i_2...i_n$. Atunci limbajul *E* va fi reprezentat în

semiautomatul A = [Z. {e. i, . i, i, $i_t\ i_2 \ldots\ i_n$, s}. *f;* e], unde

/■o, ...i,.,) .!'■
| în caz contrar ;

re. o
(s. daca *i* t, ,

f {s, i) = f (ii ... i„,i) = s, prin starea q $i_2 \ldots\ i_n$. ■

Aceluiaşi scop, şi anume regăsirea proprietăţilor limbajelor regulate cu ajutorul semiautomatelor finite, ii sînt consacrate următoarele două teoreme : **Teorema II.4.10.** *Mulţimea limbajelor regulate este închisă la produs.*

Demonstraţie. Fie E_t, E_2 e 62 (Z) şi semiautomatele A_t = [Z, SL p,sj], > = 1,2, în care limbajele E, şi E, sînt reprezentate respectiv prin mulţimile de stări S, c S^1, S_2 £ S^2.

Vom arăta că limbajul Z£$_t$ E_2 este reprezentaL în semiautomatul nedeter- minist A = [Z, S^1 u' S^2. *f;* sj] prin mulţimea de stări S_2 .

Aici :

{P (s, i)}, dacă *se* S^l | S, *f (s, i) =*

{f^1 *(s, i),* f^2 (.<g . i)}, dacă « e

{f^2*(s,* i)}, dacă s e S^2.

■> S-a notat prin U' reuniunea disjunctă.

Să notăm cu E evenimentul reprezentat în semiautomatul A prin mulţimea de stări şi fie p_1 G E_k, p_2 e E_2. Avem f^1 (,sj , pj G şi, dacă $p_2 = i_L \ldots i_n$. atunci f^2 (,s§ . *if) e f* (f^1 (*J , p_t).* ij. De aici rezultă imediat că *f* (*o • Pi P2) n S_2 + 0, adică $p_\pm$ p_2 e *E,* deci E_k E_2 c *E.*

Invers, fie p = i,... i_n s *E.* Atunci *f* (xj . *p)* n S_2 / 0, şi fie k =. min *{j* i *f* (sj , i, ... $i_{;+1}$) / f^1 (sj . ij ... i_{y+1})}. Din modul de definire a semiautomatului A. cuvîntul $p_x = i_L \ldots i_t$. e $E_}$. iar cuvîntul $p_2 = i_{k+}\ldots i_n$ s E_2. adică E G — 7*^ E_2 . ■

Teorema II.4.11. *Mulţimea limbajelor regulate esle închisă la operaţia de închidere prin produs.*

Demonstraţie. Să presupunem că limbajul *E* esle reprezentat în semiautomatul determinist A = [7. S, f;s_0] prin mulţimea de stări şi fie semiautomatul nedelerminist A' = [7, S. *F* ; .s_0], unde :

p (s i) = [^(s' ')}, dacă 5 G S \
i), f (s_0 , î)}, dacă s *e S, .*

Fie, *E"* limbajul reprezentat în A' prin mulţimea de stări S_L. Să arătăm că *E* = E'.*

Fie *p e E*.* Atunci există *k* G *N,* {I/$_L$, q_2 ,.... q_t} G *E* şi *p = q_k q_2 ... q_k* . Dar, *f* (s_0 . vJ G S, şi ca atare, dacă $q_2 = i_k \ldots i_m$. *F (s₀ , q_k* zj = *{f(s₀ , q_k* i$_x$), *f (ˢo '* h)}' S$^{e\ ve}$de imediat că *: F (s_g , q_k q_2)* n S\ / 0 şi,

mai general, *F (s₀ , h* ?2 ••• 7A) ⁿ Sj / 0, pentru *h* = i, *k.* în particular, pentru *h = k* obţinem *F (s₀* . p) n Sj / 0, deci p G *E'.*

Invers, fie p = *iₖ i₂... iₙ* G *E'.* Avem F (s₀ , p) n S\ / 0. Se vede uşor că p se poate scrie p = *qₓ q₂ ... qₖ , k* 1. unde *f* (.s₀ ; *qₕ)* G , y/ₜ = 1. fc şi deci p G £* G *E*.* ■

Semnalăm, în continuare, alte proprietăţi de închidere ale mulţimii 62 (7).

Definiţia II.4.1. *Operaţia de oglindire* esle operaţia unară în *I*,~* : 7* -» 7* dală de: p = *iᵦ i„_j ... iₗ* . unde p = *iₗ i₂ ... i„.*

Observaţia II.4.2. Operaţia de oglindire se extinde la *<Ţ>(I*)* punînd £ = = *{P* i *P*

Avem *E = E* şi *(pq) = q p.*

Teorema II.4.12. 62 *(I) este închisă la operaţia de oglindire.*

Demonstraţie. Fie *E* G 62 (7), adică / 7*/g^E l<oo, conform teoremei II.4.3. Vom arăta că | 7* / ¹ < -ţ- oo, ceea ce va antrena *E* G 62(7).

Fie *(p.q)* G g^£. Atunci V/j , *r₂* G*J*E/.ₑ (i|p r₂) = xᵦ(ʳi P'2)* ˢᵃᵘ XE *(f₃ P*
= XE *(r₃* 7 ^Vr3 • D ^e Deci & 7) ^e ^£. ■

Definiţia II.4.2. *Operaţia de închidere prin prefixe* este aplicaţia **pr :** *<Ţ>* (7*)-> -» *iŢ>* (7*) dală de

pr (£) = *{q* | 3r G 7*, *qr e* £} u {e}.

Lăsăm în seama cititorului verificarea proprietăţilor de operator de închidere (v. definiţia 1.1.13) ale aplicaţiei **pr.**

Teorema II.4.13. <2 (Z) *este închisă la operaţia* **pr.**

Demonstraţie. Fie *E* s 62 (Z) şi *A =* [Z, *S,f;s₀]* semiautomatul în care limbajul *E* este reprezentat prin mulţimea de stări Sj. Se constată imediat că evenimentul **pr** *E* este reprezentat în automatul A prin mulţimea de stări *S' = {s* | 3p S *E, p = qr, f (s₀ , q)* = s}. ■

Teorema II.4.14. *Fie E* s 62(Z)şi *A* = [Z, *S,f;s₀] semiautomatul în care E este reprezentat. Este decidabil faptul că E /0.*

Demonstraţie. Să presupunem că *E / 0.* Vom arăta că există un cuvînt *p* s *E, l (p)* < |S| , ceea ce ne va permite ca într-un număr finit de paşi (cel mult | *I*ⁿ n* = | S |). Să verificăm dacă *E 0.*

Fie *p* e *E* cu *l* (p) = min {/ *(q)* / *q* s *E}.* Dacă *l* (p) > | S atunci fie ˢi+i = , ij, *k* = 0, *l* (p), unde p = i₂... *iₙ* . Mulţimea {sₜ₊₁ | *k* = 0, *l* (p)},
avînd cel puţin | S | + 1 elemente nu poate avea toate elementele distincte. Există deci Jj, *j₂* astfel ca sᵧᵢ₊₁ = .sJi+1, 0 <*j₂* < *l* (p). Atunci

cuvîntul $q = i_t \ldots i_h\ i_{lt} \ldots$ $i_{J(}$ > s *E* şi *l (q) < l* (p), ceea ce contrazice definiţia lui p. ■

Fie Ej,..., E_r limbaje peste alfabetul $I = \{ij,\ldots, i_r\}$. Fie aplicaţia <p : *I* -» <J)* (/*) dată de

9 (e) = {«},

9 (G) = $^{E}i' J$ **= h**

9 (P7) = 9 (P) 9 (?)- Vp. 7 ^{s}

Definiţia II.4.3. *Operaţia de substituţie generată de mulţimea de limbaje M* = $\{E_x,\ldots, E_r\}$ este operaţia unară *: <Ţ>* (Z*) -»*<p* (Z*) dată de :

$$\hat{}_M(E') = \{<1\ 1\ 7\ ^{e}\ 9\ (p).\ P\ ^{e}\ E\}.$$

Operaţia de substituţie este o operaţie *(r* -p l)-ară pe *<p* (Z*) dată de :

$$(E_2 \ldots E_r, E) = <D_M(E), \text{ unde } 3Z = \{E\ ,\ldots, E_r\ \}.$$

Observaţia IE4.3. Dacă $E_s =$ unde a : 1, / -> 1, *r,* atunci aplicaţia
cp este un endomorfism pe *I** care păstrează lungimea cuvintelor şi $Q_{Jt/}(E) = = 9$ (-E)-

Teorema II.4.15. 62 (Z) *este închisă la operaţia de substituţie.*

Demonstraţie. Fie $E_r,\ldots, E_r$, *E* 62 (Z) şi $E_j = L$ ($G_;$), unde $G_j = (P_N,$ {ip ..., i_r}, *4 , F[1]), j* = 1, *r,* şi *E = L (G),* cu $G = (I_N$, {ij ,..., i_r}, ,r_0 ; F), toate gramaticile care intervin aici fiind de tip 3.

Să considerăm gramatica de tipul 3, $G' = (I_N$ u U P_N , {î,ij, ,r_0; F'), ;-I ^{r}

unde F' = J F_j u *F.*
i-i

Aici *Fj* se obţine din F, , înlocuind regulile de generare de forma rrj. —> fj , i2... ...iA prin reguli de forma .rj. -> ij *i2... i „ x'0,* dacă în *F* apare o regulă de forma .r -> ... *ih* i, ...r.. şi .iţ -> *it i2...* iA *xt ,* dacă în *G* avem regulile de generare .7- ->r. *i„ xt* , ,r(...

F se obţine din *F* înlocuind regulile de forma *x,* -> L, ... *i, x,*

sau *x,* -» ° > 1 7 n / *l*

->■ *ih ijt ... ij* cu reguli de forma *xtx'0*|.

Să arătăm că *L (G') = (EL ,..., Er, E).* Fie *p (E1 ,..., Er. E).* A lunci *p* = pj ... *ph* . unde *pk* s *Ejk ,* VA- = 1, *h* şi *ij i ih ... ijh e E.* Există atunci deducţia :

■ro =* 'h - ri => *h. -'i,.; ■■■■ii*k2 r2 => - Gi *'h - iih* prevenită prin aplicarea regulilor de generare din gramatica *G :*

-To -* G. G. - *b,,*- -ri - *'>*kl+1 - *bk, -r2.…-D -* |+1 - *iih* ■ Corespunzător, în

gramatica *G'* avem regulile de generare :

xo -To‘ -ri -To*«+1 din *F* Şi- mm *p4* e *Eh …pk* e *Ejh* apare deducţia :

$x_0^{j_1} \overset{*}{\underset{G'}{\Rightarrow}} p_1 x_0^{j_2} \overset{*}{\underset{G'}{\Rightarrow}} \dots \overset{*}{\underset{G'}{\Rightarrow}} p_1 \dots p_{j_{k_1}} x_1 \overset{*}{\underset{G'}{\Rightarrow}} p_1 \dots p_{j_{k_1}} p_{j_{k_1+1}} x_0^{j_{k_1+2}} \overset{*}{\underset{G'}{\Rightarrow}} \dots =^*$ Pi - P_{ikl}

' ' ' 12 $P'ti\ P^h$ ' d^{ec}* P^s L

G' G' o'

Invers, fie *p L (&).* în gramatica *G'* generarea $x_0 x_1 \dots x_{r-1} x_r$ există în definitiv, cinci de regulă. Presupunem că în generarea cuvîntului *p* apar toate aceste tipuri de reguli de generare. Fie, de exemplu, :

.r_0 => ,rj -> *q' => q' q"* rrjj => *q' q" rx*$_i$ => *q' q" rx% => q' q" rr'* .i| => *q' q" rr' px%=$ => q' q" rr' pp'.*

Cum în acest caz *pp' e* E_{js}, rezultă că *p* este obţinut prin substituţia ip efectuată asupra cuvîntului $i_4\ i_2\ i_3\ i_4\ i_5$ e *E.* Astfel *p* s ţp (Ej,..., E_r, *E).* ■

Teorema II.4.16. <2 (/) *este închis la endomorfismele semigrupului liber I*.*

Demonstraţie. Rezultă din observaţia II.4.3 şi din teorema II.4.15. ■

Teorema II.4.17. *Dacă E <= CQ (I) şi h esle un endomorfism al semigrupului liber I*, cme păstrează lungimea cuvintelor, atunci* **/i^{-1} (E) s (/).**

Demonstraţie. Aplicaţia Ir^1 *: I* ->■ <p* (/*) face să corespundă unui simbol *i* **e** *E* mulţimea finită Ir^1 *(i)* S /. Atunci $h^{\sim 1}$ (i) este un limbaj regulat (v. teorema II.2.1) şi Ir^1 *(E) =* (A^{-1} (ij),..., /i^{-1} (i_r), *E).* ■

în sfîrşit, limbajele regulate apar ca soluţii ale acelor sisteme de ecuaţii liniare cu coeficienţi limbaje regulate, care admit soluţie unică. Mai precis, arc loc :

Teorema 11.4.18. *Sistemul Z = ZE* u *B, in care E este o matrice nesingulară ale cărei elemente sînt limbaje regulate şi B este un vector-linie cu compo.*

nente limbaje regulate, are ca soluţie unică vectorul Z= {Zj Z$_m$) ale cărui componente sînt limbaje regulate.

Demonstraţie. Vom proceda prin inducţie asupra Ini *m* (dimensiunea ma- tricii *E).* Pentru *m* = 1, soluţia este *Z = BE** (v. consecinţa 1.6.1).
Conform teoremei II.2.3, *E** este un limbaj regulat şi din aceeaşi teoremă rezultă că şi *Z* este un limbaj regulat.

Să presupunem că teorema este valabilă pentru *in* = 1, *n* — 1 şi fie sistemul :

Z = ZE u *B, E* fiind o matrice de tip (n, n). Cum *e E$_{nn}$* (în caz contrar matricea ar fi singulară) avem :

(11.4.1) *Z^ţZ^E^u ...* u Z$_{B_}$! *E$_{n_ln}$* U *K$_n$) !■?,>„ ,* soluţia unică a ecuaţiei :

$$\%n = \hat{}nn \text{ U } (\hat{}1\ E_{ln} \text{ U } \bullet\bullet\bullet \text{ U } \hat{}n\text{-}1\ \hat{}n\text{-}1.\ n \text{ U } \hat{}_n),$$

(ultima ecuaţie a sistemului considerat). Eliminînd astfel pe Z_n din primele *n* — 1 ecuaţii, obţinem astfel sistemul :

(11.4.2) *Zj = Z$_l$ţE$_{lj}$* u *E$_{ln}$ E$_{nn}$ E$_{nJ}$)* u ... u *Z$_{n_l}$ (E$_{n_lt}$ j* u *E$_{nn}$ E$_{nj}$)* U *B„E*$_{nn}$E$_{nj}$* u *Bj , j* = 1, n — 1, în care matricea coeficienţilor este *E′* = = (E$_w$ u *E$_{t„}$ E*$_n$ E$_{nj}$)* o matrice de tip *(n* — 1, *n* - 1), avind drept elemente limbaje regulate. Matricea *E′* este nesingulară. într-adevăr, dacă *e* e *E′$_{k}$},* atunfci sau e e *E$_{k)}$* sau *e* e *E$_{kn}$* şi *e* e *E$_{nj}$* deci existenţa unui monom strict ciclic *E′.. ... E*| $_f$, care conţine cuvîntul e ar antrena existenţa unui monom strict ciclic în matricea *E,* care ar conţine cuvîntul *e,* ceea ce este imposibil, *E* fiind o matrice nesingulară.

Prin urmare, sistemul (II.4.2) admite o soluţie unică si din ipoteza inductivă componentele soluţiei sînt limbaje regulate. Din relaţia (II.4.1) rezultă că şi Z_n este un limbaj regulat. ■

Consecinţa 11.4.1. Demonstrarea teoremei precedente oferă şi un algoritm de găsire a soluţiei, algoritm care constă din eliminări succesive.

Exemplul II.4.2. Fie alfabetul {e, 1} şi sistemul :

$$(Z\hat{}Z\hat{} = (Z,Z_2Z_3)\ \{e\}\ \begin{matrix}(\{0\}\\ \{i.\\ O\\ \ldots\end{matrix}\ \begin{matrix}\text{un}\\ \{e\}\ U\ (\{e\},\\ \{1\}^*,0)\end{matrix}$$

care pe componente revine la :

$$Zj = Zj\ 0 \text{ u } z_2 \text{ U } z_3 \text{ U } \{e\}$$

$$Z, = Z_x\ 11\ U\ Z_2o\ U\ Z_30\ U\ \{lj^1*$$
$$= Z_L1\ U\ Z_2.$$

Substituindu-1 pe Z_3 în primele două ecuaţii, obţinem :

$$Zj = Zj \{0, 1\} U z_2 u \{<-'\}$$

$$Z_2 = Zj \{11, 10\} u z_2 o u \{1\}'.$$

Rezolvînd a doua ecuaţie,- găsim :

$$Z_2 = (Zj \{11, 10\} u l^*)0\backslash$$

care introdusă în prima ecuaţie ne dă :

$$z, = z_x (\{0,1\} u \{11,10\} \{()\}^* u \{1\}^* \{0\}^*),$$

adică :

$$Z, = \{1\}^* \{()\}^* (\{0, 1\} u \{11, 10\} \{()\}^*)^* = \{0, 1\}^*,$$

de unde obţinem imediat, pe Z_2 şi Z_3.

§11.5. LIMBAJE DE TIP 2 (LIMBAJE LIBERE DE CONTEXT)

Limbajele de care ne vom ocupa în acest paragraf modelează proprietăţile limbajelor de programare.

Teorema II.5.1. *Fie G = (I_N , I_T , ,r_0 ; F) o gramatică de tiptil 2. Există atunci o gramatică G_x — (I_N , I_T , a_0 ; F\), de tipul 2, astfel incit* L *(G_t) = = L (G)* \ {e} *şi G_L nu are reguli de generare de forma x -> e.*

Demonstraţie. Să construim mulţimea *F*\ . Pentru aceasta fie I_x = {.r | *x* e ^{s} I_N , *(x -+ ej* e *F}* şi $I_{j+1} = J_j$ u {.T | -T s I_x, *(x^ p)* e *Fj , p* e *Ft, j* 1. Există *k* e *N* astfel incit /j. = 7_{t+1}, deoarece mulţimea *F* este finită. Mai mult, $I_k = I_{k+V}$, Vi» s N.

•

Avem deci *x => e,* dacă şi numai dacă *x* e l_k , aşa cum rezultă din modul de formare al mulţimilor .

Alegem F_x = {(.T -> $p_±$), p_k / *e, 3 (x -*■ *p)* e *F, p_y* se obţine din *p* prin îndepărtarea simbolilor din 7_t}. Vom arăta că *L* (GJ = *L* (G) \ {e}.

Fie *p L* (Gj). Avem deducţia :

$$•^r0 => /Jj => p_2 => \ldots => p \overset{e}{} I_T,$$
$$G, \quad Ot \quad <?, \ Gi$$

sau

$_{r_0}$ => *Pi* ^{x}*Pt =* Pi ,' PÎ => P< Gt* $_{(?i}$ $_{G}$ ceea ce atrage existenta regulilor de generare de forma (;r$_{\pm}$ -> Pi, e *F*|. Corespunzător în gramatica *G* avem reguli de forma ,i -» ■ *y,* cuvîntul *y* conţi nînd pe lînga simboli din cuvîntul *:h* şi simboli neterminali din *I,* fdeci pentru care avem reguli de generare de forma ,r' -» c, care nu altereaza rezultatul final al deducţiei).

în gramatica *G* avem o deducţie de forma :

$$,r_0 \Rightarrow_{G} \text{pi .r, pj'} \Rightarrow_{G} \text{pi zp"} \Rightarrow_{G} \ldots \Rightarrow_{G} \text{p E}$$

V P = L (G) \ {e}.

Invers, să arătăm că *L(G)*| {e} s L *(Gt).* Pentru aceasta să considerăm gramatica G_2 = (/$_N$,I$_T$,.r$_0$; *F* u Fj), pentru care avem evident *L* (G) c *L (G2)* şi deci *L (G)* | {e} *Q L (G2)* \ {c}.

Rămîne să arătăm că *L* (G_2) \ {e} c *L* (GJ. Fie p E *L* (G_2) \ {e}. Atunci

în deducţia T_0 => p pot apare reguli de generare din *F* şi din F, . Pornind Gi de la această deducţie, putem construi o altă deducţie a cuvîntului p, folosind exclusiv reguli de generare din F_t , înlocuind regulile de generare din *F* cu regulile corespunzătoare din F_x (obţinute din regulile de forma (.r -» p) e *F* prin îndepărtarea din p a simbolilor neterminali din 7_{fc}). ■

Teorema II.5.2. *Fie G = {I$_N$, I$_T$, .r$_0$* **; F)** *o gramatică de tip 2. Dacăe* s *L* (G), *atunci există o gramatică G'= (l'$_N$, I$_T$, x$_0$;* F') *de tipul 2, astfel incit I,* (G) = *L* (G') *şi singura regulă de generare de forma x -> e este regula x'$_o$ -> e şi x'$_o$ nu apare in partea dreaptă a unei reguli de generare din F'.*

Demonstraţie. Fie G_r gramatica construită în demonstrarea teoremei precedente şi G' gramatica de tip 2 :
G' = (I$_N$ u {.ro}, *I$_T$* , a$_0$; Fj U -> e, 4 x$_0$}). Gramatica G' satisface con
diţiile cerute în teoremă şi *L* (G') = *L* (G_t) u {e} = *L* (G). ■

Folosind noţiuni din teoria grafurilor, vom prezenta un mod de descriere a deducţiilor din gramaticile de tip 2.

Fie G = (/jy, , .r$_0$; F) o gramatică de lip 2.

Definiţia II.5.1. *Un arbore de deducţie* în gramatica G esle un arbore care se bucură de următoarele proprietăţi :

1. — orice nod este etichetat cu un simbol din *I$_N$* u *IT* <
2. — rădăcina arborelui este etichetată cu .r$_n$;
3. — nodurile care au cel puţin un descendent propriu, sînt etichetate cu simboli din *IN* >
4. — dacă Pj, *i>*$_2$sînt descendenţi ai nodului *i>,* scrişi într-o anume ordine introdusă pe mulţimea descendenţilor nodului *u,* atunci în gramatica G există regula de generare *x -> y$_Y$ y$_2$... y$_k$,* unde *x* este eticheta nodului *i>,* iar t/y este eticheta nodului *v$_f$j* = 1, *k ;*

5.— etichetele nodurilor terminale ale arborelui (în ordinea în carfc apar) descriu cuvîntul generat prin deducţia reprezentată de arbore.

Exemplul II. 5.1. Fie gramatica G = ({x_0 , *x}, {i_1, i_2}, x_0 ; F),* unde F = = {T_0 -> i_1 xx_0 , r_0 -> ij , *x -> $x_0 i_2 x$, x -»■ i_2 i_L , x x_0* x_0}.

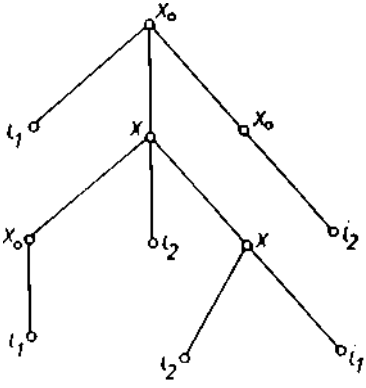


Fig. II. 5.1

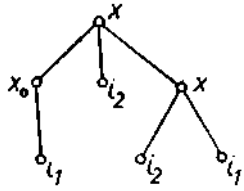


Fig. II. 5.2

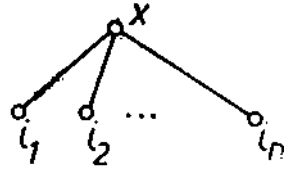


Fig. II. 5.3

Arborele din figura II.5.1 corespunde deducţiei: J'$_o$ =>$i_1 x x_0$ => i\ x_0 i_2 xx_0 => => ii ,r_0 i_2 xi_t => z\ $i_\pm$ i_2 xi_2 => i_t ij i_2 i_2 i_L i_r. ■

Definiţia II.5.2. *Un subarbore* al unui arbore de deducţie este subgraful generat de un nod şi mulţimea descendenţilor săi, păstrîndu-se etichetele din arbore. Eticheta rădăcinii unui subarbore poate fi alta decît eticheta nodului iniţial.

Exemplul II. 5.2. Subgraful din figura II.5.2 este un subarbore al arborelui din figura II.5.1. .Acestui subgraf îi corespunde deducţia : *x x_0 i_2 x => =^> i_2 x => i_1 i_2 i_2 .*

Teorema II.5.3. *Fie G = (I_N, I_T,* .r_0; *F) o gramatică de tip 2. Atunci $p^E L(G)$* \ {e}*, dacă şi numai dacă există un arbore de deducţie corespunzător deducţiei x_n => p.*

Demonstraţie. Pentru *x* <2 I_N să considerăm gramatica G_x = *(I_N,I_T, x*; F). Oricare ar fi *x* ^I_N, regulile de generare sînt aceleaşi pentru toate gramaticile G_x , deci *x => p* echivalează cu *x* => p, V.t', *X"* e I_N şi cum $G_{x>}$ = G_x, aceasta re- $G' G''_x$ vine la *x => p.*

Să arătăm mai întîi că dacă există un arbore de deducţie în gramatica G_x , ale cărui noduri terminale descriu cuvîntul *p,* atunci există deducţia *
x => p.* Demonstraţia se va face prin inducţia după numărul *n,* al

nodurilor $^{\circ}_{x}$
neterminale din arbore.

Fie $n = 1$. în acest caz arborele are aspectul din figura 11.5.3. în acest

Să presupunem că afirmaţia este valabilă pentru arborii cu cel mult n — 1 noduri neterminale şi că *p* este descris de un arbore de deducţie a cărui rădăcină este etichetată cu x şi care are *n* noduri nelerininale. /i > 1.

Să considerăm nodurile etichetate cu simbolii gj g,„ .....e /v j I_r , descendenţi direcţi ai rădăcinii x, care nu pot fi toate noduri terminale (in caz contrar arborele care descrie cuvinlul *p* ar avea un singur nod nelerminal). Avem, evident, *(x-> $y_k y_m$)* e *F.* Fie *{y_h.... y_{Jr}} =* {y, ... y_m} n *1_K.* Nodurile *(y_{jh} / , h*1, /} constituie rădăcinile unor subarbori. care au *= n* - 1 în cel mult vîrfuri nelerminale. Fie $g_L q$, cuvintele descrise de aceşti subarbori. baza ipotezei inductive, avem în gramaticile G„, ... *G...* deducţiile $^{1}Ji_k$ =* <7» ■ /i = 1, *i-* Este clar că dacă *h < k,* desccndenlii nodului $g_{>;}$ (care

X descriu cuvinlul *q_h)* apar la stingă descendenlilor nodului y_t (care pe $c/_t$). Ca atare, avem în gramatica G_x deducţia x =) descriu y_k... y_m

JG, - y_{iT} - *Um* => 'i -- 7r - *'n = P-* $^{1|}$nde {i_r,..., i,.} = {g> ... n I_T y_m}

Reciprog, să presupunem că în gramatica G_x există deducţia x => *p.* Vom arăta că există un arbore de deducţie în gramatica G_x care descrie cuvîntul *p.* Vom proceda inductiv după lungimea *n* a deducţiei de mai sus. Dacă *n* = 1, atunci (x -* *p)* e *F* si există un arbore al cărui aspect este redat în figura II.5.3.

$_e$Să presupunem că afirmaţia este adevărată pentru *n* — 1 şi că deducţia x => *p* are lungimea n. Fie x -» g_t... y_m primă regulă de generare utilizată.

Esle clar că orice simbol din *p* trebuie, fie să aparţină mulţimii {y .....y_m} n 0 I_T , fi$^{e\ s}$ă poată fi dedus dintr-un simbol din mulţimea {g_i,..., y_m} 0 *1_N* . Astfel gy, subcuvîntul din *p,* care se deduce din y_j, se află la stînga subcu- vîntului q_k , care se deduce din y_k pentru *j < k.* $_t$

Astfel putem scrie *p = p_k p_2 ...., p_n* , unde *p,* ^{E} I_T *U* {r/$_1$ ...., q_m} şi y_f => *Pj.*

^{G}*vj*

Vj = 1, *m.* Prin ipoteza inductivă există un arbore de deducţie în gramatica Gj, care descrie subcuvîntul p_t. Arborele obţinut prin conectarea rădăcinilor acestor arbori la nodul x va conslitui arborele de deducţie care descrie cuvîntul *p.* ■

Noţiunea de arbore de deducţie poate fi utilizată în demonstrarea următoarei teoreme :.

Teorema II.5.4. *Este decidabil faptul că limbajul generat de o gramatică de tip 2 este vid.*

Demonstraţie. Vom construi un algoritm prin care să se determine dacă un limbaj generat de o gramatică de tip 2 este nevid. Fie $G = (I_N , I_T , x_0 ; F)$

•

o gramatică de tip 2. Să considerăm deducţia $x_0 => p$ şi arborele corespunzător. Vom ataşa acestui arbore un arbore de deducţie dintr-o mulţime finită de arbori. Să presupunem că există un drum în arbore cu două noduri v_t şi i'., etichetate cu acelaşi simbol y situate la înălţimile n_l şi n, respectiv, cu

*

n, > rij . Subarborelui cu rădăcina in ii corespunde 'o deducţie $y => p_l$,

•

iar celui cu rădăcina în v_2 deducţia $y => p_2$. Să observăm că p_2 trebuie să fie un subcuvînt al lui p_x , eventual chiar p_A . Cuvînlul p se poate scrie sub forma $p = p_3 p_t p_A$, unde p_3 sau p_4 pot fi eventual cuvînlul nul. Dacă înlocuim subarborele cu rădăcina în Pj prin acela cu rădăcina in v_2 obţinem un cuvînl

*

$p_3 p_2 Pi$ (eventual chiar cuvînlul p) şi $.r_0 => p_3 p_2 p_A$. In acest fel am eliminat cel puţin un nod P din arbore. Dacă noul arbore are un drum cu două noduri etichetate la fel, aplicăm acelaşi procedeu cuvintului $p_3 p_2 p_4$. Procedeul sc poate aplica de un număr finit de ori, deoarece fiecare pas elimină cel puţin un nod din arbore şi arborele are un număr finit de noduri. Arborele rămas după epuizarea aplicării acestui procedeu conţine în fiecare drum noduri etichetate cu simboluri dinstincte, deci are drumurile de lungime cel mult « egale cu j I_K u I_T| . Conchidem că dacă $L (G) =£ 0$, există o deducţie $,T_0 => p$.

a p

e *L (fi)* căreia îi corespunde un arbore de deducţie avind drumurile de lungime cel mult | l_x u 1_T |. Cum mulţimea arborilor de deducţie ale căror drumuri au lungimea cel mult I I_N U I_T I este finită, putem afirma că dacă L (G) / 0 există un arbore de deducţie aparţinînd acestei mulţimi finite şi astfel într-un număr finit de paşi se poate deduce valabilitatea propoziţiei .fi (G) / 0". ■

Vom studia în continuare unele proprietăţi ale limbajelor de tip 2, legate de aşa-numitele forme normale.

Teorema II.5.5. *Fie $G = (I_X , J_T . ,r_0 ; F)$ o gramatică de tip 2 cu L (fi)* / 0. *Există o gramatică echivalentă* $\mathbf{G_t}$ **,** *astfel că pentru orice x* s I_N *să existe o de- duclie x => p.*

Gi

Demonstraţie. Ataşăm fiecărui simbol *x<^ I_N* gramatica *G_x = (J_N, I_T , x ; F).* Dacă *L (fi$_x$)* = 0, atunci putem elimina *x* din *I_N* şi toate regulile de generare din *F* de forma *x -» q* sau *x_L ->* ... ,r ..., fără a modifica limbajul generat de gramatica G. Anume, fie G_L gramatica astfel obţinută din G, *G_L = (I'$_v$, I_T, r_0;F').*

Să arătăm că *L* (GJ = *L (fi).* Este clar că *L (Gf)* c *L* (G), deoarece orice deducţie în *G_J* este o deducţie în G.

Invers, să arătăm că *L* (G) c *L (GJ.* Vom proceda prin reducere la absurd. Să presupunem că există *p* e *L* (G) şi *p L* (GJ. Atunci există o de-

•

ducţie de forma *x_0 => p_4 xp_2 => p* E *L (G), x<^ I_y | I'$_N$.*

•

Astfel trebuie să existe o deducţie de forma r => *q* s *L (G_x),* ceea ce contrazice faptul că *L (G_x)* = 0. ■ G

Observaţia 11.5.1. Teorema precedentă ne permite să eliminăm din alfabetul iniţial al gramaticii toate simbolurile care nu apar efectiv în deducţie.

Consecinţa Il.-j.l. I-'ie *G* o gramatică de tip 2 cu *L (G) => 0.* Atunci există o gramatică *G' = (I'$_N$, I_r , ,T_0 ; F')* de tip 2, cu *L (G') = L (G)* astfel inc îl

•

V.r G /A. , , există o deducţie de l'orma ,r_0 => p: ,rp_3 => *p_l p_2 p_3 , p_t , p_2 , p_3 e I'$_T$. Demonstraţie.* Fie G_r = *(I$_N$, I_T , x_0 ; F)* o gramatică care generează pe *
L (fi) şi satisface condiţiile din teorema precedentă. Dacă .r_0 => *q_x xq.$_t$, q_i , q_2 e= (h*

« •

G (/A, (J *I_T)*.* atunci există o deducţie ,r_0 => *p_L xp_3 => p_Y p_2 p_3* G *F_T ,* deoa- (?1 Gr'j

rece pentru Vy G *I_N* există o deducţie *y => p I*$_T$* . Fie *l'$_N$* mulţimea simbo- (?i

linilor *x* cu proprietatea că există o deducţie *x_0 => q_y xq$_2$.* Această mulţime (7i

o putem construi efectiv. Astfel, ,r_0 G *I'$_N$* ; apoi adăugăm mulţimii orice simbol care apare în partea dreaptă a unei reguli de generare *x -» q* dacă *x* aparţinea deja mulţimii, procedeu finitar, deoarece numărul de reguli de generare este finit.

Mulţimea *F"* se obţine prin eliminarea regulilor de generare care conţin simboli din *I_N | I'$_N$* . Este clar că *L (G') = L(G).* ■

Definiţia II.5.3. O *deducţie imediată la stingă* a cuvîntului *p* G L **(G), G** fiind o gramatică de tip 2, este o deducţie de forma :

x => ft => ft> => ••• 7» => P> unde *ft = ^{r}k ^{u}i > ^{r}i ^{e}* > FT E (JN U *I_T)** Şi

(ft T ft) $^{s\ F}$> ft *(JN* U I_T)**,* iar ft+i = >*j* ^{l,}*j"j* ■

în mod analog se introduc *deducţiile imediate la dreapta.*

Observaţia II.5.2. Unei deducţii imediate la stînga (la dreapta) îi corespunde în mod biunivoc un arbore de deducţie.

într-adevăr, plecînd de la un arbore de deducţie obţinem o deducţie imediată la stînga (la dreapta), aplicînd sistematic substituirea cu prioritate a primului (ultimului) simbol care apare în descendenţii unui nod oarecare. Reciproca este evidentă.

Teorema II.5.6. *într-o gramatică de tip 2 pentru orice cuvînt p* G *L (G) există o deducţie imediată la stînga.*

Demonstraţie. Vom proceda prin inducţie asupra lungimii *n* a deducţiei,

•

arătînd că pentru orice deducţie ,T_0 =S> *p* există o deducţie imediată la stînga prin care se obţine cuvîntul *p.*

Dacă *n* = 1, avem *(x₀ -» p)* G *F* şi deci deducţia imediată la stînga x_0 => *p.*

Să presupunem afirmaţia adevărată pentru *n < k* şi fie *x₀* => *q₃* => *p* o deducţie de lungime A + 1 cu *q₃* = i/j ... y_m , y$_;$ G I_N u I_T *, j = 1, m.* Putem

•

scrie *p = i\ r₂... r_m* , unde y_t => r_j *, j* = 1, *m.*

Prin ipoteza inductivă există deducţii imediate la stînga de la *Pj* la r$_;$, *j* = = 1, *n.* Dacă y^ G I_T , atunci zj = y_j .

T)acă i/j E I_N avem deducţia imediată Ia stingă x_0 => /y_1 ... y_m => r_t y_2... y_m. In continuare vom utiliza pentru deducerea cuvîntului *i\ r₂ y_a... y_m* deducţia imediată la stingă a cuvintului r_2 din y_2*,* obţinînd astfel deducţia la stingă :

•l'o => *yi* ■■■ *'Jrn => 'L* y_2 ■■■ *Um* => '1 '2 - *Vm* ^{m}- ^{d}- ■

Definiţia II.5.4. O *gramatică G de lip* 2 este *ambiguă* dacă există un cuvînt *p* E (G) pentru care avem cel puţin două deducţii imediate la stingă distincte, pornind de la acelaşi simbol neterminal.

în caz contrar *G* este *neambiguă.*

Exemplul II. 5.3. Fie gramatica *G* = ({x_0 , 2}, {i_r , i_2}, .r_0 ; {x_0 -> i_l , ,r_0 -» -> ij i_2 x_g i_2, x_g -»■ ij *xi₂, x -»i₂* x_0 ; *x* -> i\ *xxi₂).* Atunci pentru cuvîntul i_t i_2 i_t i_2 E *E (G)* există următoarele două deducţii imediate la stingă :

•^{r}o **1 *2* ^{T}o **2* '1 **2 *1* ^{Z}2 '

şi

$$\bullet r_0 \Rightarrow ij\ xi_2 \Rightarrow 1'2\ i\% \Rightarrow <1 <2\ '1\ G\ '$$

deci gramatica G este ambiguă.

Observaţia II. 5.3. Gramaticile neambigue sînt importante în special pentru procesele de compilare a programelor scrise în limbaje de programare.

Exemplul II. 5.4. Fie $G_t = (\{x_0\}, \{i\}, x_0 ; \{x_0 \to x_0\ x_0 , x_0 \to i^5\})$ şi $G_2 = = \{(\%)> \{'\}.\ Ho ; \{.Vo\ i^5{}'\ Ho^{,-5}\ l/o\})$-

Atunci $L(GJ - L(GJ = \{i^{5fc} \mid k\ 1\}$. Gramatica G_t este ambiguă deoarece pentru cuvîntul i^{15} avem deducţiile imediate la stînga :

$$x_0 \Rightarrow x_0\ x_0 \Rightarrow s> x_0\ x_0\ x_0 \Rightarrow i^5\ x_0\ x_0 \Rightarrow i^{10}\ x_0 \Rightarrow i^{15},$$

Şi

$$x_0 \Rightarrow x_0\ x_0 \Rightarrow i^5\ x_0 \Rightarrow i^5\ x_0\ x_g \Rightarrow i^1\text{»}\ x_0 \Rightarrow i?^5,$$

în timp ce gramatica G_2 este neambiguă.

Acest exemplu arată că,pentru acelaşi limbaj pot exista atît gramatici ambigue cit şi nearnbigue care să-l genereze. De aceea are sens definiţia :

Definiţia II.5.5. Un *limbaj* este *inerent ambiguu.* dacă toate gramaticile care-1 generează sînt ambigue. în caz contrar (adică în cazul existenţei unei gramatici nearnbigue care generează limbajul) acesta se zice *neambiguu.*

Teorema II.5.7. *Dacă L-, , j = 1, k sînl limbaje neambigue disjuncte două k cile două, atunci limbajul* $\bigcup_{i-i}$ *este un limbaj neambiguu.*

Demonstraţie. Vom presupune pentru simplitate $k = 2$, demonstraţia pentru $k > 2$ făcîndu-se la fel.

Fie $Gj = (1\#,. I_T , x_{oj}$; FJ, $L_f = L(G_f)$, $j = 1, 2$ două gramatici neambigue cu acelaşi alfabet terminal, Lj n $L_2 = 0$. Putem presupune că $(f_{vi} \setminus \setminus {}^{7}T)\ 0\ (Ava \setminus Z_r) = 0$. Fie $G = (I_N. I_T, x_0$; F), unde z_0 G I_{N1} J $<_{V2}$. IN = Avi u A1V2 U {Jol ' I' = *1* U *1'2* U {*^0 -* ^01 ' '*0 -* ^02 J • (^) ” = 7., u $J._2$. Fie ,$T_0 \Rightarrow$ p, $\Rightarrow$ ... $\Rightarrow$ p, şi $x_0 \Rightarrow$ </, ... $\Rightarrow q_t$ două deducţii imediate la stînga ale cuvînlului $p_s = q_t$ în gramatica *G.*

Deoarece in F sîntnumai două reguli de generare de forma $x_0 \to z.$ şi anume .r_0 -» ,r_{01} şi ,r_0 -» _r_{02} . urmează că $p_x . q_t$ e {z_{01} , ,r_{02}}.

Să presupunem că $p_t = q_k = z_{01}$. Deoarece $(I_{N1} \mid I_T)$ n $\{I_{N2} \setminus f_T)$

= 0. aici regulile de generare,folosite în ambele deducţii provin din F_1 şi cum G_k era neambiguă, cele două deducţii coincid. în mod analog deducţiile

coincid
dacă $p_t = q_k = .r_{02}$.

Să presupunem că $p_t = T_{01}$ şi = .r_0., . Atunci deducţia ,r_0 => p_9 include *a*

numai reguli din l'_k , deci $p_{„}$ G , iar .r_0 => q_t include numai reguli din *c*

F_2 , deci q_t L_2 .

Prin urmare cuvîntul $p_s = q$, *1^ C| L2* = 0, ceea ce este imposibil. Deci cazul $p = x_{ol}$, $q_k = x_{02}$ nu poate apare şi gramatica *G* este neambiguă.®

Teorema II.5.8. *Pentru orice gramatică G de tipul 2 există o gramatică echivalentă G_t care nu conţine reguli de generare de tipul x_k -> x_2, x_k,* **.r2** e *IN •*

Demonstraţie. Fie $G = (I_N , I_T , x_0$; F) şi F = F' u *F",* unde F' este mulţimea regulilor de generare de forma x_t -* .r_2, x_k, x_2 s I_N şi F" = F \ F'.

Construim gramatica $G_k = (I_N , I_T , x_0$; FJ unde F_k se obţine din F prin înlocuirea regulilor de generare din *F* în modul următor :
dacă (z, -» x_2) e F_k şi {(x_2 -> $w_{\#}$) | wy *IN}* £ F înlocuim (jj -» .r_2) cu mul ţimea de reguli {(z_2-> Wy)}.>

Să arătăm că *L* (GJ = *L* (G). Se vede imediat că *L* (GJ £ *L (G).*

Fie p s *L (G).* Avem z_0 => p. Conform teoremei 11.5.6 există o deducţie *G* imediată la stînga :
•To => ?i =* Î2 => - => = *P,* unde *q, =* /yZy u_{jt} q_{i+1} = *rp^u,, r, e* F_T , »y .
1 '/(IN U *IT)** Şi *vj} s F-* Dacă *(xj* -» pj s *F", j* = 1, *m,* aLunci *(Xj->*

Vj) s Fj şi p <s L (GJ. Fie *k* cel mai mic indice pentru care (z_t- > G F' şi presupunem că *(xs -> vf)* e F' pentru *k < h,* iar *(xk -> vh)* s F". Atunci cuvintele q_k , $q_{k+1}q_{lt_1}$ au toate aceeaşi lumgime şi cum deducţiile sînt imediate la stînga, simbolii înlocuiţi ocupă aceleaşi poziţii, adică deducţia q_k => q_{k+1} => ... => q_{h_1} are de fapt aspectul r_t p_t u_t => r_k x_{k+1} u_k => => ... => r_k x_{lt_1} Üj. , ceea ce implică q_k => r_k v_h u_k = q_h , deducţie obţinută prin utilizarea unei reguli din Fj. Astfel p G *L* (GJ şi *L* (G) = *L* (GJ. ■

Teorema II.5.9. *(teorema lui Noam Chomskg de existentă a formei normale). Orice gramatică G de tip 2 cu e L* **(G)** *esle echivalentă cu o gramatică G_k de tip 2, in care regulile de generare sînt de forma x* ij ,r_2 *sau x -> i, x* ;r_1, x_2 G e /A. , i s Z_T .

Demonstraţie. Fie *G* o gramatică de tip 2. Conform teoremei 11.5.8 există o gramatică echivalentă G_t = *(J_N , I_T* . r_0 ; *F)* de acelaşi tip, care nu convine reguli de generare de fomra rr, x_2 .ij . x_2 G J_N .

Astfel, dacă *(x-> y)* G *F,* atunci *y* G I_T şi regula de generare are forma *x -» i,* admisă în teoremă.

Fie *(x -> y_Y y_2... y_m)* G *F, m >* 2, i/J = {$_{?71}$,..., y_m} n I_T şi ....r_t} o mulţime de *k* simboluri. Să considerăm o nouă mulţime de reguli de generare *F'* obţinută prin adăugarea la *F* a regulilor de forma *-h y_{jh}* (care sînt de forma prescrisă) şi înlocuirea regulii *x -> y_1 y_2... y_m* prin *x -> i>$_l$ ... i>$_m$,* unde :

$$p. = I \text{ ' dacă } lJi \,\&\, /jv \;/\; Zj, \text{ dacă } y\hat{} \text{ G } J_T$$

Gramatica G, = *(F_N , I_T , x_0* : F'), unde 4 se obţine prin adăugarea la /v a tuturor simbolilor z_n . este echivalentă cu gramatica G_2, deci cu gramatica *G.*

într-adevăr. dacă $p \overset{*}{\Rightarrow} g$. atunci $p \overset{*}{\Rightarrow} g$. deci L ($G_t$) £ /, (G.,). Vom arăta *Gi* t?i

prin inducţie relativ la lungimea unei deducţii că dacă *x => p,* unde *x* I_N

£ F_N , p e F_T , atunci *x => p.*

Afirmaţia este evidentă pentru deducţii de lungime 1.

S-o presupunem valabilă pentru deducţii de lungime cel mult *k* şi fie

$x \overset{*}{\Rightarrow} p$ o deducţie de lungime *k* + 1. In mod necesar, prima regulă de genc- G.
rare utilizată esle de forma *xi| ... v_m , m* 2. Putem scrie *p = p_t ... p_m,*

unde $v_J \Rightarrow Pj$ *.j* = 1, *m.* este o deducţie de lungime cel mult *k.* Dacă *i>j* G
G»
\4, atunci există o singură regulă de generare din *F',* pe care o putem folosi şi anume v_J *-»i_f , i,* G I_T . Jn acest caz *= pj .*

Din modul de a-1 defini pe F', rezultă existenţa unei reguli de generare în F de forma *x -» y_1 ... y_m* , unde

Vj, dacă *Uj* G I_N *ij,* dacă v_J *s* r_N| I_N

Pentru *Vj* s J_N deducţia $\Rightarrow p_f$ are lungimea cel mult *k* şi prin ipoteza c,
inductivă, avem $P_j \Rightarrow p$, . Prin urmare $x \Rightarrow p$ şi $L(G_2)$ £ F (GJ.

Să observăm că G_2 nu are forma impusă de enunţ, deoarece conţine reguli de generare de forma r -> $t_1\ t_2 \ldots t_m$, $m > 2$, *tj* s I_N ,

Construim o nouă gramatică $G_s = (I\#\ ,\ I_T\ ,\ ,r_0\ ;\ F'')$ astfel : pentru fiecare regulă de generare (.r $t_2 \ldots t_m$) e *F', m* 3 introducem noi variabile *iij* . $u_2\ u_{m_2}$ şi înlocuim $x \to t_Y\ t_2 \ldots t_m$ prin mulţimea de reguli de generare {.r -> /j u_1, u, $t_2\ u_2, \ldots, u_{m_2}\ t_{m_2}\ u_{m_2}$, u_{m_2} -> - Ui U; 4 se obţine prin adăugarea la I'_N a simbolilor u_j , iar *F"* se obţine din *F'* prin înlocuirea de mai sus.

Avem $L(G_2)$ $L(G_3)$, deoarece, dacă $x \Rightarrow p$ atunci .r => *p.* Invers, să G« *G3* arătăm că $L(G_3)$ c $L(G_2)$. Fie $p\ s\ L(G_3)$. Vom proceda prin inducţie asupra lungimii *l* a deducţiei r => *p.*

Pentru cazurile *l* = 1 şi *l* = 2 avem evident *p* e (G_2). Să presupunem că pentru toate deducţiile r => *p,* de lungime cel mult *k* există deducţia Ga $y \Rightarrow p$, unde *y* coincide cu .r dacă $x\ I'_N$ şi fie $x \Rightarrow p$ o deducţie cu lungime G_2 *Gs* $k + 1$. Prima regulă de generare din această deducţie este obligatoriu de forma $x \text{-»} u >_2$, unde iv_i , w_2 e J".

Dacă w_1 , w_2 e]'N regula de generare $x \text{-»} u >_t w_2$ aparţine lui *F',* din gramatica G_2 şi deci există deducţia $x \Rightarrow Wy\ w_2 \Rightarrow p$ = p, p_2 , unde $w_1 \Rightarrow p_x$, a_3 *a,*

* $w_2 \Rightarrow p_2$ sînt deducţii de lungime cel mult *k.*

Aplicînd ipoteza inductivă, există deducţiile $\Rightarrow$ pj , $tv_2 \Rightarrow p_2$, adică G_2 GJ $x \Rightarrow$ p.

Să observăm că în prima regulă de generare, aplicată în deducţia considerată nu putem avea *x* e Rămîne încă posibilitatea ca prima regulă de generare să fie de forma $x \to w_1\ u$, unde iu_1 e I'_N , e F_N .
Aceasta implică existenţa în G_2 a regulii $x\ w_Y \ldots\ w_n$ şi în G_3 a regulilor .r -> $i_{V1}\ u$, . $u \hat{}\ w_2\ u_2, \ldots, u_{_3} \Rightarrow w_{n_2}, u_{n_2} \to u >_n \hat{}\ u/_n$.

Cum $iv_t \Rightarrow$ p, cuvîntul p se poate scrie p = $p_L\ p'_x$, unde => p_t, p_a p_a

$u_L \overset{*}{\Rightarrow}$ p'. Conform ipotezei inductive $w_1 \overset{*}{\Rightarrow} p_t$. Apoi, deoarece singură re- Gi Gg gulă de generare care-1 conţine în partea stingă pe este $u_1 \to w_2 u_2$, p[se poate scrie $p\{ = p_2 p_2$, unde $w_2 \overset{*}{\underset{G_i}{\Rightarrow}} p_2$, $u_2 \overset{*}{\underset{G,}{\Rightarrow}} p_2$.

Analog avem $w_2 \overset{*}{\underset{G,}{\Rightarrow}} p_2$ şi $p_2 - p_3 p_3$ etc.

Astfel $Wj \overset{*}{\Rightarrow} pj$ _ j = 1, 11, unde $p = p_t \ldots p_n$ şi deci $x \overset{*}{\underset{G_t}{\Rightarrow}} p_Y \ldots p_n = p$. ■ (7>

Consecinţa II.5.2. Dacă drumurile unui arbore de deducţie pentru gramatica *G* sub forma normală a lui Chomsky au lungimea maximă egală cu *j* cuvîntul terminal corespunzător are lungimea cel mult egală cu 2^{j-1}. Dacă pentru *p* s *E* (G) avem *l* (p) > $2^{>-1}$ atunci intr-un arbore de deducţie pentru cuvîntul *p* există - măcar un drum de lungime cel puţin *j,* cum rezultă imediat ţinînd seama de forma specială a regulilor de generare.

Exemplul II. 5.5. b'ie G_t = ({x_0 , x_4 , x_2}, {i_4, i_2}, x_0 ; {x_0 -> i_2 xy, x_0 -* *ij* x_2, ^{x}i '1 , - '1 ^{x}o ' *1 -> '2 -^{r}i ^{T}i. $^{x}2$ -* '2 < $^{x}2$ - '2 ^{x}o ' $^{x}2$ 'i $^{X}2$ *2})' ^{0} gramatică de tip 2. Ne propunem să găsim gramatica G_2 de tip 2 în forma normală a lui Noam Chomsky, echivalentă cu G_4.

Regulile de generare care au deja forma admisă sînt x_4 -» i_4 , x_2-> i_2. Pentru celelalte reguli de generare vom aplica construcţia indicată în demonstraţia teoremei, aşa cum rezultă din tabelul de mai jos.

Regula de generare din G_4	Regulile de generare corespunzătoare din G_2
^{x}o | * ^{X}	XQ ~* *Zj* xy , *Zj*
^{x}o | ' $^{x}2$	x_0 -» z_2 x_2 , z_2 -» i
^{X} | ' ^{X}	x_4 -> z_3 x_0 , z_3 ->
X —| ^{1} ^{X} X1	^{x}i -» z_4 *Xi Xi,* z_4 -»i_2
x_2—| ^{X}	$^{x}2$ ^{x}o ' ^{z}s -* ^{f} 2
• —' ■^{T} x_2	$^{X}2$ $^{z}6$ $^{X}2$ $^{X}2$ ' $^{z}6$ (1

в continuare vom construi gramatica G_3 înlocuind regulile de generare Xj -» z_4 *Xj Xj* şi x_2 z_6 x_2 x_2 respectiv prin x_4 -> z_4 *Uj,* u_1 ->

x_4 x_4 şi x_2 -» —* -6 ^2 ’ ^2 $^{X}2$ $^{X}2$ •

Teoremele care urmează constituie rezultate preliminare în vederea prezentării unei ‘alte forme normale pentru gramaticile de tip 2 şi anume forma normală datorată lui Sheila Greibach.

Teorema 11.5.10. *Fie G = (I_N , I_T , x_0 ; F) o gramatică de tip 2. Fie* (x -> -> /y *x' r2)* s *F şi* {x' -> *t/j*, x' -» *q2,..., x' -> </„} F toate regulile de generare care conţin x' în partea stingă. Atunci L* (G) = *L* (G_4), *unde* G_k = (I_N , I_T, x_0 ; F_t) *Fj fiind obţinut din F înlocuind regula de generare x-> zy x' r2 prin mulţimea de reguli x ->■ /y q{ r2 ,* x -> iy *q2 r2 ,...,* x -> /y *qn r2 .*

Demonstraţie. Avem *L* (G_4) *L* (G) deoarece, dacă regula x -» /y *q} r2* esle folosită intr-o deducţie din G_4 alunei x => *iy x' r2=) ly q3 r2 .*

G G

Invers, să arătăm că *L* (G) c /, (GJ. Dacă în deducţia *x => p* se utilizează $\tilde{G}$ regula .r -+ *x' r2* atunci in mod neceşar se utilizează ulterior şi regula *x' -> qf* şi aceşti doi paşi pot li înlocuiţi printr-unul singur, anume *x => r}* r/3 r_2. ■ *G,*

Teorema 11.5.11. *Fie G = (I_N , 1_T . .r_0 ; F) o gramatică de tip 2 şi F' =* = {.r -» *xpj* ./ = 1, n}, *F"* = {.r -+ q_t |j = 1, m} *mulţimile de reguli de generare care conţin x în partea stingă, F' fiind constituită din acele reguli în care x apare şi ca primul simbol din partea dreaptă. Atunci L* **(G) =** *L* **(G_1),** *unde* $G_{\pm}$ = (/v u {4’ I_T ’ r o ’ ^i)’ *fiind obţinut din F prin înlocuirea mulţimilor F' şi F" respectiv prin F[şi Ff , unde :*

$$/\bullet; = \{:\text{-}\text{»}/>,. \qquad pj\ 7 = 1, n\}.$$

si

$$= \{.c\ q,, x\hat{}\ q_f\ z,\ 7 = 1, m\}.$$

Aici pj e *l’T* , 7 = 1, n, q_n e F_T , *h* = 1, *m.*

Demonstraţie. în baz^ teoremei II.5.6 putem utiliza pentru deducerea cuvintelor din limbajele considerate numai deducţii imediate la stînga.

Fie *p* **e** *L (G).* Dintr-o deducţie imediată la stînga a cuvîntului *p* în gramatica G putem construi o deducţie a lui *p* în gramatica G_x astfel : fiecare apariţie în deducţia imediată la stînga a şirului

$$\text{'i } xr_2 \underset{G}{=>} r_t\ x_{Pjl}\ r_2 \underset{G}{=>} i\ j\ xp_{jt}\ p_{jk}\ r_2 \underset{GG}{=>} \ldots \underset{G}{=>} x_{Pj}\ {}_{Pji}\ r_2 \underset{G}{=>}$$

$$\overset{\text{'1 } \text{<ii } Pj_h\text{- } Pj, \text{ '2 >}}{\underset{G}{\text{''}}}$$

va fi înlocuită prin :

$$' 1 -^{7} 2 \quad G, \quad '1\ (li\ Pi\ h\ "2^{=>} \bullet\bullet\bullet "i\ Qj\ Pj_{h} \bullet\bullet\bullet Pj, \sim^{\prime}2 \quad G, \quad G, G, \quad C,$$

$$=> 'j\ 'li\ Pi_{„} - Pj.\ Pi, '2 \bullet \quad G,$$

şi astfel L (G) L (GJ.

Să observăm că deducţia din gramatica G_x nu este o deducţie imediată la stingă.

Să considerăm acum o deducţie imediată la stînga a lui *p,* în G_t . înlocuim şirul :

$$'1\ .1/_{2} \underset{G\ i}{=>} r_t\ q,\ zr_2 \underset{Gi}{=>} r_{\pm}\ p_{jh}\ zr_2 \underset{Gi\ Gi}{=>} ... => i|\ q,\ Pj_h \ldots p_{Jt}\ zr_2$$

$$=^{*} '1\ <li\ Pi_{„}- Pi.\ Pi,\ '2- \quad G,$$

din această deducţie prin

$$r,\ xr_2 => \underset{\mathbf{G}}{r_x}\ xp_h\ r_2 \underset{a}{=>} r_k\ xp_h\ p_h\ \underset{GG}{r_2} =* \ldots => r_k\ xp_j \ldots$$

$$\blacksquare\blacksquare\blacksquare\ Pi,\ Pi,\ '2 =^{*}_{G} '1\ <li\ Pi_h - Pi, \blacksquare$$

obţinînd astfel o deducţie a lui *p* in L *(G).* Astfel L (G_t) c *L* (G). ■

Teorema II.5.12. *(teorema lui Sheila Greibach de existenţă a formei normale). Orice limbaj E de lip 2 cu e E poate fi generat de o gramatică in care orice regulă de generare este de forma x -> ir, unde x & I_N , i & I_T , r I'_N.*

Demonstraţie. P'ie G = ($/_v$, 1_T , x_0 ; *F)* o gramatică de tip 2 sub forma normală a lui Chomsky şi 1_N = {.r_0 , .ij ..... .r„}. Vom construi efectiv o gramatică care să satisfacă la condiţiile cerute şi care să genereze limbajul *L* **(G),** procedînd in mai multe, etape.

Etapa I. Vom modifica regulile de generare astfel incit dacă *(x_f x_k r)* E E *F, r* E F_N să avem *k j,* obţinînd o gramatică G_k, echivalentă cu G.

Pentru aceasta vom proceda inductiv asupra indicelui /, *j* = 1, *n,* pentru *j* = () condiţia fiind de la sine îndeplinită.

Să presupunem că au fost modificate toate regulile de generare de forma *Xj -> x_k r* astfel îneît *k j* pentru *j* = 1, *li* şi gramatica obţinută în acest stadiu să fie echivalentă cu G.

Vom modifica acum regulile de generare'care au in partea stingă variabila x_{h+1}. Dacă .r_{ft+1} -» x_k *r* este o regulă de generare cu *k < li* + 1 atunci, utilizînd ipoteza inductivă şi teorema II.5.10, vom înlocui această regulă cu o mulţime de reguli de forma .r_{ft+1}

-> x_m *r,* proces care se poate repeta pînă cînd *m* > /i + 1 (de cel mult de *h* — 1 ori).

Etapa a Il-a. Dacă există reguli de generare de forma x_f -> ,r$_{\#}$ *q* vom aplica teorema II.5.11 mulţimii de reguli de generare care au în primul membru pe x_j obţinînd o gramatică *G.,* echivalentă eu Gj . ale cărei reguli de generare au una din formele :

$$Xj \text{ -> } x_k\, r,\ k > j,\ 1 \text{ e } J_N ,$$

$$x_j\, ir,\ i \text{ E } I_T,\ r \text{ E } F_N ,$$

$$'i < b < 1 \text{ S } (/jV \text{ U } \{?i \text{ -'„}\})^{*}\bullet$$

Etapa a 111-a. Vom modifica regulile de generare astfel ca primul simbol din partea dreaptă să fie un simbol terminal.

Observăm mai întîi că toate regulile de generare care conţin în partea stingă pe x_n au ca prim simbol în partea dreaptă un terminal. Apoi, primul simbol din partea stingă a regulilor de. generare care încep cu ,r„$_{_1}$ poate fi sau x_n sau un simbol terminal. în primul caz aplicînd teorema II.5.10 putem înlocui regula respectivă cu noi reguli de generare care au ca prim simbol în partea dreaptă un simbol terminal. Vom proceda similar pentru .r„$_{_2}$ ele.

Etapa a IV-a. în sfirşit. înlocuim regulile de generare care au *z* in partea stingă şi primul simbol din dreapta nelerminal prin reguli in care primul

simbol din partea dreapta este un simbol terminal utilizînd în acest scop teorema II.5.10.■

Construcţia din această teoremă se poate urmări în :

Exemplul II.5.6. Fie gramatica de tip 2, $G = (\{x_0, x_x, x_2\}, \{z'j, i_2\}, x_0 ; \{x_0 \to x_L\ x_2,\ x_1 \to x_2\ x_0,\ Xj \to i_2,\ x_2 \to .r_0\ x\backslash,\ x_2 \to r\backslash\})$. Urmind construcţia descrisă în demonstraţia teoremei precedente ne propunem să găsim forma normală de tip Greibach a gramaticii *G.*

Etapa I. Să observăm că toate regulile de generare care au în partea stingă x_Q sau $x_\pm$ satisfac condiţia cerută. Rămîne să ne ocupăm dc regulile care încep cu x_2, anume $x_2 \to x_0\ x_1$ care va fi înlocuită pe rînd cu

Xg Xj Xg Xi ,

şi apoi prin :

$$x_2 \to x_2\ x_0\ x_2\ x_1 \text{ şi } x_2 \to i_2\ x_2\ x_1 .$$

Obţinem astfel gramatica G_1 echivalentă cu G, $G_\pm = (\{x_0, x_x, x_2\}, (ij, i_2\}, {}^{X}O\ '\ K\ {}^{-T}1\ '\ a\text{-}l\ \text{“*}\ {}^{-T}2\ {}^{X}0\ '\ {}^{X}1\ (2\ '\ {}^{a}2\ \text{“*}\ {}^{X}2\ {}^{X}0\ {}^{X}2\ {}^{X}1\ '\ {}^{X}2\ {}^{!}2\ {}^{X}2\ '\ {}^{X}2\ \text{“*}\ {}^{!}1\})$-

Etapa a Il-a. Vom aplica teorema II.5.11 regulilor de fi

x_0 - x, x_2	$x_2\ z_2$ *X,* Xj	. ,$r_0\ x_2$ Xj z
•1'1 -> -r_2 X_o	x_2 -> z\	Z -» $x_o\ x_2$ Xj
Xj -> i_2	X $_2$ “* ^'$_2$ ${}^{X}2$	
	${}^{X}1\ {}^{0}/_{0}$ *2 z1 z	

Etapa a IlI-a. regulile de avînd x_2 în partea stingă ca prim simbol un terminal. Apoi, vom folosi aceste reguli de generare în regula x_v -+ $x_2\ x_0$ în aşa fel încît primul simbol din partea dreaptă să fie un terminal şi în sfîrşit, folosind ultimele reguli obţinute vom modifica regula x_0 -» $x_x\ x_2$. Obţinem astfel

$$x_2 \to i_2\ x_2\ x_1 ;\ x_2 \to i_1 ;\ x_2 \to i_2\ x_2\ x_1\ z ;\ x_2 \to i_1\ z ;$$

$$x_1 \to i_2 ;\ x_1 \to i_2\ x_2\ x_1\ x_0 ;\ x_1 \to i_1\ x_0 ;\ x_1 \to i_2\ x_2\ x_1\ zx_0 ;$$

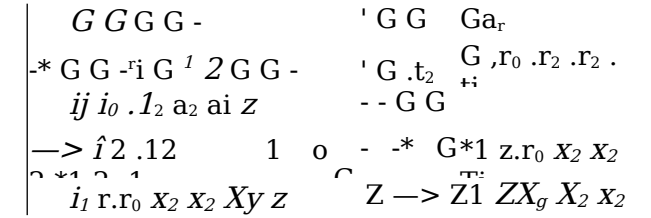
$$x_1 \to i_1\ zx_0 ;\ x_0 \to i_2\ x_2 ;\ x_0 \to i_2\ x_2\ x_1\ x_0\ x_2 ;\ x_0 \to i_1\ x_0\ x_2 ;$$

x_0 ~* fg Xg $x_t\ zx_0$ **Xg j** XQ

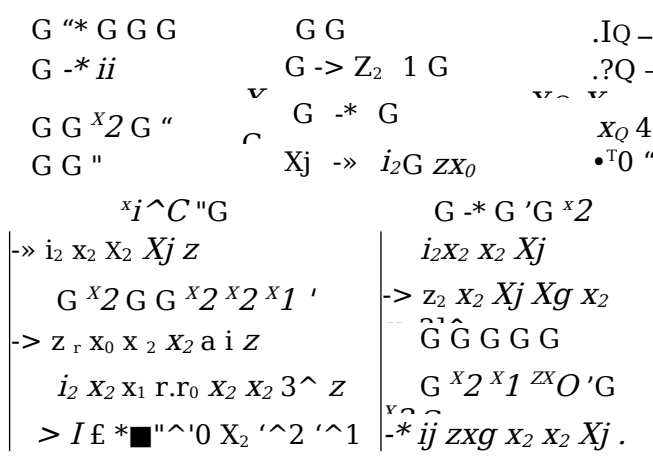
generare care începe cu x_2 (întrucîl apare regula de generare $x_2\ x_2$,...), introducînd simbolul z şi înlocuind regula de generare $x_2 \to x_2\ x_Q\ x_2\ x_x$ prin z -> $x_0\ x_2\ x_t$ şi z -> $x_0\ x_2\ x_L$ z, iar regulile x_2 -» $i_2\ x_2\ x_i$, x_2 -> ij prin x_2 -» $z_2\ x_2\ x_\pm$, x_2 -» $i_2\ x_2\ x_L$ z, x_2 —►i_1,x_2 ¯» z'j z.

Mulţimea de reguli de generare pentru noua gramatică va

Etapa a IV-a. Regulile de generare care conţin în partea stingă pe *z* au ca prim simbol în partea dreaptă pe x_0 . Folosind regulile de generare care conţin pe x_0 în primul membru, cele două reguli z -» x_0 x_2 x_1 *z* şi z —> x_0 x_2 x\ voi’ fi înlocuite respectiv prin :



Gramatica sub forma normală Greibach, echivalentă cu gramatica dată
G este deci G' = ({.r_0 , Xj . x_2, -}, {z\, i_2}, z_0 ; F'), unde F' este alcătuit din următoarele reguli :



Ne vom ocupa în continuare de unele proprietăţi de închidere ale limbajelor de tip 2 (v. şi §.II.2).

Teorema 11.5.13. *Clasa limbajelor de lip 2 este închisă la operaţia de substituţie.*

Demonstraţie. Fie E\,..., E_r, *E* limbaje de tip 2 asupra alfabetului *1* şi $E_j = L(G_f)$ unde $G_j = (P_x$, {z\i_r}, x^*_0 ; FJ, $j = 1$, *r* şi $E = L$ (G) cu G = $(I_N$, {z\ i_r }, x_0 ; F). Putem presupune, fără a restrînge generalitatea că P_N n $Iy = 0$ pentru j / k şi I_N n $F_{„} = 0$, $j = 1$, *r.*

Ne propunem să construim o gramatică de tip 2 care să genereze limbajul $E_r](E)$ obţinut prin operaţia de substituţie generată de r-uplul de limbaje $\{EjE_r\}$ (v. definiţia II.4.3).

Fie $G_o = (I_N$ u U ' {'i . . 'r}> G î *F)* cu F_o = U U F' unde F' se obţine din F prin înlocuirea în fiecare regulă de generare a

simbolilor terminali *ij,..., i_r* respectiv ;rj.rj .

Să arătăm mai întîi că ……(E) £ *L (G_o).*

Dacă *q* <5^..£$_r$} (^) există *p* e *E* astfel ca *q* = 9 (p) (q> fiind aplicaţia utilizată în definiţia II.4.3). Fie deci *p = i_{ia} , i_{ja}... i_{jn}* unde j_{jlc} e {1, 2,..., /•), *k* - 1, *n.* Urmează că *q = q_{jt} q_{ja} ... q^* unde q_{jk} e E_{jk} , *k* = 1, *n.* în gra- matica G_o avem deducţia .r_0 => .rj[1] *xfr* ... .rj» => q_h *xfc ... xyt => ... => q.*

G_o Go Go

Invers, fie *q* e *I.* (G_o) adică .r_0 => *q.* Există atunci un cuvînt *y* e [,r*, .r, G.

* *

.... .TJ}*, *y* = .r£‘ ...a’J" astfel încît să existe deducţiile x_0 => *y* =& *q* şi

G, G,

*

x_0=>i> = ij. ij, ... i_{in} ■

G

■***k***

Aceasta revine la a scrie *q = q_{ja} q_{ja} ... q_j* cu ajc -> q_{jk} , *k* = 1, n.

Aslfel ^{G}>*^{k}

q = o (p) (v. definiţia II. 1.3) şi *q* s ……(/:)•■

Teorema II.5.14. *Clasa limbajelor de tip 2 este închisă la endomorfismele semigrupului liber I*.*

Demonstraţie. Deoarece orice limbaj finit este limbaj regulat, deci şi dc tip 2, afirmaţia se obţine ţinînd seama de faptul că dacă *Ej =* } unde a : 1, r-»l, *r,* atunci aplicaţia din definiţia II.4.3 este un endomorfisin pe *I** şi <t>$_v$ (E) = *? (E)* şi se utilizează şi teorema precedentă.■

Teorema II.5.15. *Dacă E este un limbaj de tip 2 şi h este un endomorfisin al semigrupului liber I* care păstrează lungimea cuvintelor atunci 1C[1] (E) esle un limbaj de tip 2.*

Demonstraţie. Aplicaţia *Ir[1] : 1* — <p (I*)* face să corespundă unui simbol *i* e *I* mulţimea finită *Ir[1] (i)* c *I.* Atunci *Ir[1] (i)* este un limbaj regulat şi *lri(E) = ţ ţh-[1] (ij,..., h~i (i_r), E). ■*

Teorema 11.5.16. *Clasa limbajelor de tip 2 nu-i închisă la intersecţie şi com- plementariere.*

Demonstraţie. Vom arăta că există E_1,E_2 s Jj_2 şi E_1(^E_2^^$_2$. Fie gramaticile :

Gi = (ta > -ii}- {'1, G - '$_3$}, -[l]o; {■’■<) ->• -[r]o '3 > -[r]o -* -[T]i '3. *1 '1 '2 ’

[x]i “ h [x]i *2})>

Şi

G_2 = ({.r_0 , .TJ, *{ir , i.>*, i_3}, .r_u ; {.r_0 -* i_Y ,r_0 , ,r_0 -+ i_t , .r_x i_2 i_3 ,

■' 1 '2 x'i '3})•

Avem *L* (G,) = {<" i" *i™* j *m, n* 1} şi *L* (G_2) = {i™ i_2 i_3 / *m, n* 1}. Evident *E = L* (GJ *L (*G_2*)* = {(" i" i'j j *n*]}. Să presupunem că *E* e *J>*$_2$. Există alunei o gramatică de Lip 2, G = *(*I_x , {z'j , i_2, i_3}, .T_0 ; F) şi = *L (G).*

Fie *x* / ,r_0 , *x e* I_N şi *x* =*> p* e {ij, *i*„, z_3}*. Dacă această deducţie este unică *G* considerăm gramatica *G' =* {/$_v$ \ {a}, {i_r, *i.*,. î$_3$|, .r_n : *F')* unde *F'* se obţine din *F* îndepărtând toate regulile de generare de forma .r -» *u* şi înlocuind fiecare apariţie a lui *x* în partea dreaptă a unei reguli de generare din *F* prin cuvîntul *p.* Evident *L (fi) = L (G').*

Deci putem presupune că pentru orice simbol nelerminal (in particular

şi pentru ,r_0) există cel puţin două deducţii de forma *x* =*> p* e {z_r ♦ , i_2 , i_3|*. G

Fie *k* maximul lungimii cuvintelor care apar în partea dreaptă a unei astfel de deducţii. Pe de altă parte, cum zf^{+1} i*$^{+1}$ i_3^{+1} e *E = L (fi)* există deducţia

,T_0 =*> pj *xp*$_2$ =*> pj z/p_2 = iţ$^{+1}$ î$_2^{+1}$ ij^{+1}. Astfel *(x -» q)* e *F, l (q) < k* şi mai există *G* încă cel puţin un cuvînt terminal *q' / q* astfel că *x* =*> *q'*. Prin urmare există *G* un subcuvînt *q* al cuvîntului i*$^{+1}$ f|$^{+1}$ i*$^{+1}$ cu *l (q) k* şi *q' q* astfel încît cînd înlocuim pe *q* prin *q'* în cuvîntul i*$^{+1}$ i_2^{+1} i_3^{+1} cuvîntul rezultat aparţine limbajului *E,* ceea ce este evident imposibil, deoarece acest cuvînt nu are forma prescrisă. Prin urmare *E* £*_*$_2$.

în consecinţă clasa £$_2$ nu este închisă nici la complementariere deoarece în caz contrar fiind închisă la reuniune ar rezulta şi închiderea la intersecţie. ■

Teorema II.5.17. Fie *E* e <£$_2$ • *Există* ,n_2 *e N astfel Incit, dacă p* s *E şil* (p) > *atunci p = tuquw, cu l (uqv) n*$_2$*, u şi v nu pot fi ambii egali cu e astfel incit pentru orice* /i > 0 *să avem : tu*^{h} *qv*^{h} *w* e *E.*

Demonstraţie. Fie *G = (*I_N , I_T , x_0 *; F)* o gramatică de tip 2 înforma normală a lui Chomsky^şi *E = L (G).* Dacă = *k* atunci fie = 2^{t_1} şi n, = 2^{fc} .

Fie p G *E* şi *l* (p) > 2^{t-1}. Conform consecinţei II.5.2 în orice arbore de deducţie pentru cuvîntul p există un drum de lungime cel puţin *k.* Să considerăm intr-un asemenea arbore un drum *de' lungime maximă.* Acest drum va conţine două noduri , s_2 astfel încît :

1) Nodurile au aceeaşi etichetă *x.*

2) Nodul Sj este mai apropiat de rădăcină decît s_2.

3) Drumul de la Ia nodul final (conţinut obligatoriu de acest drum de lungime maximă) este cel mult de lungime *k* + 1.

într-adevăr, din ultimele *k* + 2 noduri ale drumului de lungime maximă numai nodul terminal este etichetat cu un simbol din I_T . Celelalte *k* + 1 noduri sînt etichetate cu simboli din I_N şi cum I I_N I = *k* în mod necesar două noduri au aceeaşi etichetă şi aceste două noduri satisfac condiţiilor 1) —3).

Atunci, subarborele cu rădăcina în nodul Sj reprezintă o deducţie a unui subcuvînt pj de lungime cel mult $2^k = n_2$.

Dacă subcuvîntul p_2 a rezultat în urma deducţiei descrisă de subarborele cu rădăcina în s_2 atunci pj = $r_r\ p_2\ r_2$ unde *i*| şi r_2 nu pot fi concomitent egali cu cuvîntul nul deoarece prima regulă de generare folosită în subarborele cu rădăcina în este de forma $x\ x_2$, .Tj , ,r_2 G I_N .

Avem

$$T \overset{\bullet}{=>} i|\ x\overset{\bullet}{r_2} => /\blacksquare;\ p_{„}\ r_2 = p, , c \qquad c$$

unde / *(i*| p_2 r_2) 4: $2^k = n_2$. De aici rezultă că pentru orice *Ii* > 0 avem :

$$x => \underset{G}{/t}\ xi2 = \underset{G}{>} rt\ p_2.$$

Cum cuvînlul *p* se poate scrie $p = lp_l\ w = li|\ p_2\ r_2\ w$ rezultă că afirmaţia făcută este adevărată punînd *u* = , *v* = r_2, *q* = p_t . ■

Observaţia II.5.4. Teorema precedentă permite ca pentru limbajele de tip 2, infinite pornind de la un anumit cuvînt *p* din limbaj să construim cuvinte din limbaj, oricît de lungi. Din acest motiv această teoremă este cunoscută sub denumirea de „Ierna de pompare" (v. Arbib [1]).

Teorema 11.5.18. *Problema finitudinii limbajului general de o gramatică G de tip 2 este decidabilă.*

Demonstraţie. Fie n_r, n_2 numerele definite în teorema precedentă şi *p* G e *L* (G) cu *l* (p) > . Atunci $p = tuqvw$ şi $tu^h\ qv^h\ w$ G *L* (G), V/i > 0. Deci

l (u) + l (o) > 0.

Prin urmare, dacă există *p* G *L* (G) cu *l* (p) >• n_x atunci *L (G)* este infinită.

Să presupunem acum că *L* (G) este infinită. Atunci există un

cuvînt oricît de lung în *L (G),* în particular există p G *L (G)* cu *l* (p) > *n₂*, p = *tuqvw* cu *l (uqu) < n₂, l (u) + l* (a) > 0 şi *tu^h qv^h w* G *L (G), Vh* 0.

în particular, pentru *li* = 0 avem *tqw L (G)* şi *l (tqw) < l (tuqvw)* şi *l (tqw)* > nx. Dacă / *(tqw) > n±* 4- *n₂* repetăm procedeul pînă cînd găsim un cuvînt din *L (G)* de lungime Z cu Hj < *l* rij + *n₂*. Astfel *L (G)* este infinit dacă şi numai dacă există p G *L (fi)* cu *nr<j.l (p)* < n1 4- *n₂*.

Cum numărul acestor cuvinte este finit putem decide într-un număr finit de paşi dacă *L (G)* este finit sau nu. ■

§ II. 6. CARACTERIZAREA LIMBAJELOR DE TIP 3 ÎN CLASA LIMBAJELOR DE TIP 2

Se ştie (v. observaţia 11.2.2) că orice limbaj de tip 3 este şi de tip 2, iar existenţa unor limbaje de tip 2 care nu sînt limbaje regulate va rezulta din acest paragraf, în care ne propunem să facem individualizarea limbajelor de tip 3 în clasa limbajelor de tip 2.

Definiţia II.G.l. *O gramatică de tip 2, G = (I_N , I_T ,* ,r0 ; F) *are proprietatea de autoimersiune* dacă există *x* s astfel incit ,r s *pxq,* unde *p, q* s *e Un* U /T)*\ {e}.

Un limbaj de tip 2 are proprietatea de autoimersiune dacă el este generat **de** o gramatică de tip 2 cu proprietatea de autoimersiune.

Observaţia II.6.1. O gramatică de tip 3 nu are proprietatea de autoimersiune. într-adevăr, ţinînd seama de definiţia II.2.1, deducţiile într-o grama-

• • lică de tip 3 au forma ,r => *px₁* , unde .T, Xj e /A. , *p e 1^,* , sau *x => p* cu x S Av > /' e *U •*

Vom arăta printr-un exemplu că există gramatici de tip 2 cu proprietatea de autoimersiune.

Exemplul 11.6.1. Fie gramatica de tip 2, G = ({x0 , a^}, {ij, i2}, *x₀ ; F),* unde *F* = {x0 -> *i_x x₀* i2, *x₀ , x₁ -> i₂}.* Se observă că G este cu autoimersiune, deoarece există regula *x₀ -> x₀ x_L i₂.*

Teorema II.6.1. *Un limbaj de tip 2 este şi de tip* 3, *dacă şi numai dacă nu are proprietatea de autoimersiune.*

Demonstraţie. Conform observaţiei II.6.1 gramaticile de tip 3 nu au proprietatea de autoimersiune.

Invers, să presupunem că gramatica G = *(I_N , I_T , x_0 ; F)* este de tip 2 şi nu are proprietatea de autoimersiune.

Putem presupune că oricare ar fi *x* s /ᵥ există *p, q* s *ţI_N* ţj 1_T)*

astfel incit să existe deducţia $x_0 \Rightarrow pxq$ deoarece în caz contrar renunţarea la *x* şi la deducţiile care încep cu *x* nu modifică *L (G).*

Să presupunem mai întîi că oricare ar fi *x* E I_N există *i*| , r_2 s (/ₙ ţj I_T)* astfel ca să avem o deducţie $x \Rightarrow r_x\ x_0\ r_2$. Să observăm că în acest caz regu- *G*
lile de generare care conţin un simbol neterminal în_partea dreaptă pot avea una din formele :

(11.6.1) *x^-pyq*

(11.6.2) *xpy*

(11.6.3) *x^qy*

(11.6.4) *x^y*

unde *x,y^I_N, p,q^(I_N|J I_TY* | {e}.

Dacă *F* conţine o regulă de generare de forma (II.6.1) atunci se obţine deducţia :

$$X \Rightarrow pyq \Rightarrow_G PP1\ x_Q q_x q \Rightarrow_G \text{ppj}\ p_2\ xq_2\ q_\pm\ q,$$

deci G are proprietatea de autoimersiune, ceea ce contrazice ipoteza.

Dacă *F* conţine simultan reguli de tipul (II.6.2) şi (II.6.3) alunei obli- nem o deducţie de forma :

$$\bullet^r \Rightarrow py \Rightarrow_G PP_L\ {}^*0\ 71 \Rightarrow_G PP1\ Pi\ 7_2\ 71 \Rightarrow_G PP1\ Pi\ y'\ 7\ 72\ 71$$

$$\Rightarrow_G PPi\ Ps\ Pa\ {}^x o\ la\ 'Pli <h \Rightarrow_{<:} PPi\ Pi\ Pa\ P_A\ {}^x 'G\ 'la\ 'l'li\ 'Ii \bullet$$

ceea ce de asemenea implică existenţa proprietăţii de autoimersiune.

Dacă *F* ar conţine o regulă de generare de forma (II.6.2) atunci cuvîntul *p* s F_T deoarece în caz contrar *F* ar conţine reguli de tipul (II.6.1) sau (II.6.3). ceea ce antrenează proprietatea de autoimersiune.

Conchidem că în cazul considerat *F* conţine reguli de

generare de forma ;r -> *py* cu *p* e F_T sau ,r -» *y,* deci *G* este de tip 3.

Dacă *F* ar conţine o regulă de generare de tipul (II.6.3) (fără a conţine şl reguli de generare de forma (II.6.1) sau (II.6.2) deoarece atunci *G* ar avea proprietatea de autoimersiune) atunci *q* F_T . într-adevăr, dacă *q* ar conţine şi simboli neterminali această regulă ar apare ca o regulă de tip (II.6.1) sau (II.6.2).

Fie deci (x -» *yq)* e *F* cu *q* s I'_T . Atunci limbajul *L* (G) (obţinut prin operaţia de oglindire) este de tip 3 şi deoarece clasa limbajelor de tip 3 este închisă la operaţia de oglindire şi limbajul *L* (G) = *L* (G) este de tip 3.

Limbajul *L* (G) este de tip 3 şi în cazul în care *F* nu conţine reguli de generare de forma (II.6.1—3).

Să presupunem acum ca există *x* s f_N astfel îneît să nu apară nici o de- * ducţie de forma $x \Rightarrow l \mid x_0 r.,.$ Vom demonstra prin inducţie după $n = \mid I_N \mid G$ că *L* (G) esle de tip 3.

Pentru n = 1, *L (G)* este de tip 3 deoarece neexistînd deducţii de forma L* $x_0 \Rightarrow r_1\ x_0\ r_2$, *F* conţine un număr finit de reguli de generare de forma x_0 -+*p, p* e l'_T şi deci | *L* (G)| < oo .

Să presupunem afirmaţia adevărată pentru n *= h* şi fie Kv I = /< + 1. Considerăm gramaticile $G_1 = (I_N \setminus \{x_0\}, I_T, x ; l'\mid)$ unde F_r *se* obţine din *F* prin eliminarea regulilor de generare care conţin pe x_0 şi $G_2 = (J_N \setminus \{x\}, I_T$ u $\{x\}, x_0 ; F_2)$, unde F_2 se obţine din *F* prin înlăturarea tuturor regulilor, care conţin pe *x* în partea stingă (adică G_2 se obţine din G considerîndu-1 pe *x* ca un simbol terminal).

Limbajul *L* (GJ este de tip 3 prin ipoteza inductivă, iar *L* (G_2) este de acelaşi tip aşa cum rezultă din prima parte a demonstraţiei. Avem L(G) = = *L* u (L (G_2) 6 Lr)> unde *L* este rezultatul substituţiei lui *x* în *L* (G_2) prin

într-adevăr, dacă *p e L* **(G₂)** p *I'T* avem evident *p* s *p* (G), jar *p P* alunei *p* = / j p_k /•, p_2 ... r_k , unde .r_t => *i*| xr_2 *x* ... xr_k şi *x* => ... xr_k . şi ,r_x => *p*, ...r_t => p_t. Deci avem în *G* deducţia .r_0 => -> / j *Pi r„ x* ... xr_k => *i*| *p, r„ p.*, ... p_k„*i* r_k = *p.* de unde *P* u *(I. (G)* 0 *I'T) <= <= ((')*■

Dacă *p* e *p ((;)* şi în deducţia ,r_0 => *p* nu apare simbolul T alunei r_n => *p* şi *p* s *L (Gj)* f| *I*T* . Dacă în deducţia ,r_0 => *p* apare la un moment dat *x G* această deducţie se poate scrie

$$\bullet' \text{ o} \Rightarrow 71\ {}^{T}7a\ * - {}^{T}<7_{(} \Rightarrow p,$$

unde *rp . j* = 1, *t* nu conţin pe *x* .şi regulile de generare utilizate în deducţia

z$_0$ => (fa T $q_{t>}$ T ... xq_t nu au *x* în partea stingă. Prin urmare ,r_0 => q_t $xq_{t>}$ *x* ... xq_t . *G* <?2

în continuare, dat fiind că nu există deducţii de forma *x* => r_k buie aplicate în mod necesar reguli de generare care fac parte din *F*, şi deci avem *p* = q_k p_k q_2 p_2 ... p_{t_k} q_t , unde *x* => *p*, ,..., *x* => p_{t_2} .

Deci *p* s *L* şi *L (G)* £ *L* u (L (G_2) n F_T).

Limbajul *L* u (L (G_2) D *I*T)* este un limbaj de tip 3 în baza proprietăţilor de care se bucură clasa de limbaje de tip 3. ■

Observaţia II.6.2. Teorema precedentă dă o caracterizare a limbajelor regulate în clasa limbajelor de tip 2.

Observaţia II.6.3. Există incluziunea strictă J?$_2$ z> *Jţ$_3$* cum rezultă din teorema precedentă şi exemplul 11.6.1.

Vom încheia acest paragraf punînd în evidenţă o proprietate de închidere relativă la familiile *Jj$_2$* Ş* -£$_3$ •

Am arătat (v. teorema II.5.16) că familia de limbaje *J>$_2$* nu este închisă la intersecţie. Are loc, însă :

Teorema II.6.2. *Intersecţia unui limbaj de tip* 2 *cu un limbaj de tip* 3 *este un limbaj de tip* **2.**

Demonstraţie. Fie E_k e *J>$_2$* şi E_2 s *Jj$_3$* limbaje peste acelaşi alfabet I_T. Fie G_k = (I_N , I_T , x_0 ; *F)* gramatica de tip 2 cu E_k = *L*

(G_x) şi $A_2 = (1_T$, S, $f; s_0)$ semiautomatul în care limbajul E_2 este reprezentat prin mulţimea de stări S_2 s S. Să presupunem mai întîi că *e* E_k şi *e* E_2 . Conform teoremei II.5.1 putem presupune că *(x -» e) F,* Vz s . Fie $S_2 = \{s^1,..., s^{ft}\}$ şi limbajul reprezentat în semiautomatul A_2 prin mulţimea $\{s^f\}$, *J* — 1, *h.* Este elar că E, n $E_2 = E_Y$ 0 (U 2) = U $(\hat{}1$ H $E^j{}_2)$.

j-i *i-i*

Limbajul Ej n FJ_2 este generat de gramatica de tip 2, $G' = ((I_N$ u $I_T)$ x *x S x S,* I_T , (.T_0 , s_0 , .s^j); F") unde *F'* constă din toate regulile de generare avind una din următoarele forme :

a) (x, s, $') -> (i/t, *s, sQ* $(y_2$, s_t, $s_2)$ ... $(y_{„}$, s_{n_v} *s'),* unde .s-, ,s', *s,* G S, *j* = 1, n - 1 şi (x-> i/j 1/, ... $y_{„})$ G F, y_j G Z_v u /r,

b) *(x,* s. s') x, unde x G I_T şi *f* (,s-, *x)* = s'.

Fie p G E_x n EJ_2. Atunci x_0 *p* şi *[* $(s_0$, *p)* = s^1. Fie *p* = i_2... i_n, $^s n = f$ (-'o' '1 '2 ••• '*)> A = 1, n — 1. Vom arăta că *p* G *L (G*).* Intr-adevăr,

♦

putem considera deducţia $(r_0$, s_0 , *s')* => $(i_t$, s_0 , $(i_2$, s_1, $s_2)$ ... $(i_n$, s_{n-1}, $s^j)$

G^

din gramatica G^1. Avem de asemenea regulile de generare $(i_1$, s_0 , s,) -» ,...

..., $(i_n$. .s„_,, $s^j)$ -> i_n. Aplicîndu-le deducţia de mai sus obţinem deducţia (x_0, s_0, ?) => *p,* adică *p* e *L* (G^j).

Invers, fie *p e L* (G*). Prima regulă de generare utilizată este (x_0 , s_0 , $*) -> -> (l/i, ^{s}o' ^{v}i)(l/2> ^{s}i> $^{s'}$z) ••• *(Mic'* ^{s}L-i > ^{s}it)- continuare vom aplica numai regulile de generare indicate în tipul a). Vom obţine în mod necesar deducţia (x_0. s_0. ?) => (i_1, SQ , sj (i_2, Sj, s_2)... (i_n $s^{1'}$). Pentru a obţine cuvîntul *p*

G>

este obligatorie utilizarea regulilor de forma b) şi anume: (ij, s_0 , Sj) -» i_x,..., ..., ($i_{„}$, , , s^f) -> i_n . Existenţa acestor reguli de generare implică *f* (s_0 , i_x) = .s_p /•($!, i_2) = s_2,..., *f* (s,,!, i_n) = ?, deci *p* = i_2... *i,* e Ej 0 $E^j{}_2$.

Astfel *L (&)* = Ej n $E^j{}_2$ şi deci E_r n E_2 este un limbaj de tip 2. Cum clasa limbajelor de tip 2 este închisă la reuniune rezultă că E_L n E_2 este un limbaj de tip 2.

Dacă numai unul din limbaje conţine cuvîntul nul atunci construim o gramatică G' de tip 2 cu simbolul iniţial x_0 care generează limbajul $(E_l$ | {e}) D (E_2 \ {e}) aşa cum am procedat în prima parte a demonstraţiei si Ej 0 $E_2 = L$ (G').

Dacă e G Ej n E_2 (adică ambele limbaje conţin cuvîntul nul)

atunci adăugăm gramaticii G' un nou simbol iniţial x'_o şi regulile de generare x_0' -> *e* şi x^-> x_0 . Noua gramatică generează limbajul Ej (1 E_2. ■

§ II. 7. LIMBAJE DE TIP ALGOL

Limbajele utilizate în „conversaţia" cu maşinile electronice de calcul (aşa-numitele limbaje de programare), sînt intermediare între limbajul ştiinţific utilizat în mod curent de om şi „limbajul-maşină“ utilizat în interiorul

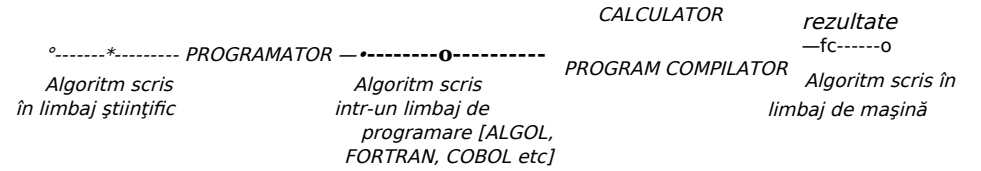


Fig. II. 7.1

calculatorului, motiv pentru care împrumută caracteristici ale ambelor tipuri de limbaje.

Sarcina traducerii unui text scrie în limbajul ştiinţific (care descrie algoritmul rezolvării unei probleme) în limbajul de programare revine programatorului.

Sarcina traducerii textului scris în limbajul de programare în limbajul- maşină revine calculatorului însuşi care o execută cu ajutorul unui program special numit compilator[1]). Textul astfel prelucrat este iar preluat de calculator şi executat, maşina oferind în final rezultatele problemei.

Schema acestei activităţi este reprezentată în figura II.7.1.

Deoarece traducerea din limbajul de programare în limbajul-maşină se efectuează de către calculator rezultă că limbajul de programare trebuie să posede o descriere precisă, conţinută într-un număr finit de reguli.

Aceste reguli se pot da în parte în forma *definiţiilor de tip Backus* numite şi *definiţii recursive* care sînt de fapt ecuaţii pe mulţimea unor limbaje (înţe- legînd aici prin limbaj o parte a unui semigrup liber). Astfel, limbajele care sînt soluţii ale unor sisteme de ecuaţii abstractizează proprietăţile limbajelor de programare.

Pentru descrierea unui limbaj de programare (adică pentru formularea definiţiilor despre care a fost vorba) se foloseşte un limbaj mai larg numit *metalimbaj.* Simbolurile care fac parte din acesta le vom numi *simboluri meta- lingvistice.* Acestea sînt :

1) litere mari şi mici ale alfabetului latin
2) cifrele arabe
3) parantezele obişnuite „(“ şi „)“
4) parantezele unghiulare şi „>“ care servesc la delimitarea cuvintelor care descriu noţiuni ale limbajului de programare
5) simbolul : =“, care are semnificaţia, „este definit ca“
6) simbolul „ | “ care are semnificaţia „sau“.

Iată un exemplu de definiţie în metalimbaj :

<literă> : : = *A* I *B* j *C* / *D* / *E* / *F* / *G* / *FI* / *I* j *J* / *K* / *L* / *M* I *N* / *O* I *P* / *Q* / *11* / s | *T* / *u* | v | w ; x | y | *z* /

< cifră > : := 0 1 | 2 | 3 | 4 | 5 | 6 | 7 ; 8 | 9

Vom da acum definiţia a două noliuni comune unor limbaje de programare (ALGOL, FORTRAN etc.)

< identificator > : : = < literă > | < identificator > < lileră > ' j < identificator > < cifră >

< întreg fără semn > : : = < cifră > | < întreg fără semn > < cifră > Fie alfabetul *1* = {A, *B,* Y. *Z,* 0, 1,..., 9}. Să considerăm limbajele = {.1,....Z} şi E_2 = {0,...,9}. Fie X şi Y limbajele, alcătuite respectiv din identificatori şi întregi fără semn. Definiţiilor de mai sus le corespund ecuaţiile :

$$* = Ei \text{ U } -$$

$$Y = E_2 \text{ tj } YE_2.$$

Trecem acum la prezentarea formală a noţiunilor descrise mai sus.

Fie $I = \{I_J, \ldots, y_m\}$ un alfabet finit şi F_a Q I^* o mulţime finită de limbaje. Să considerăm alfabetul $K = \{y_1,\ldots,y_m, 7?_n\}$.

Definiţia II.7.1. Un *term* este un cuvînt din *K*.* Simbolii dintr-un term sînt fie cuvinte din semigrupul *I* coeficienţi ai fermului,* fie simboli de forma *Ej , j* = 1, *n variabile ale fermului.*

Limbajul descris de un term $T = p_k\ E_{jt}\ p_2\ E_{jt}\ \ldots\ E_{jf}\ p_{t+1}$ esle limbajul $\{Pi'L - >'i_tP_{tl}\hat{} \mid {}^ri_k\hat{}^Ei_k\hat{}\ A = 1\sim\}$, pe care-1 vom nota cu $T\ ({}^Er\ {}^E{}_{„})$.

Definiţia II.7.2. O funcţie *li : (Ţ)*$_f$ *(I*)*^{n} *— <J)*$_f$ *(I*) ')* se numeşte *funcţie standard peste alfabetul 1,* dacă :

$$h\ (E, .\ E_{„}) = \underset{i\text{-}I}{\text{tj}}\ T_f\ (E, \ldots.. E_{„})$$

T_f fiind termi din *K*,* avînd ca variabile limbaje finite.

Ne vom ocupa în continuare de sisteme de ecuaţii de forma :

$$\begin{aligned} {}^Er &= H_1\ (E_{,,}\ldots,\ E_{„}) \\ E_n &= H_n\ (E_{\pm}..E_{„}) \end{aligned} \qquad \text{(II.7.1}$$

unde $H_j\ (E_1,\ldots,\ E_n)$ sînt funcţii standard peste alfabetul *I.*

Folosind notaţia vectorială sistemul (II.7.1) se poate scrie sub forma :

(II.7.2) **E=H(E),**

unde **E =** (E_x,..., *E„),* **H(E) =** *(H1 (E1 H„* (E_x ,..., E„))

Se spune că **H** este o funcţie vectorială standard dacă toate componentele sale sînt funcţii standard.

Teorema 11.7.1. *Dacă* **E £ E' .st H** *esle o funclie vectorială standard alunei* **H (E) £ H (E'),** *adică funcţiile vectoriale standard sini monotone.*

Demonstraţie. Pentru simplitate vom demonstra afirmaţia în cazul cind **II** are o singură componentă, deoarece demonstrarea în cazul general revine la acelaşi raţionament aplicat fiecărei componente în parte.
Aslfel

$$\mathbf{H(E) =} 11(7\text{-}, ,..., E_n) = \mathrm{U}\ T_k\ (E_k ,..., E_n) \bullet = \mathrm{U}\ T_k\ (/\text{-}'' ,..., E'_n) =$$

$$= \mathrm{H}(\mathrm{E};,...,7:_n) \mathbf{= H(F/)}$$

aceasta, deoarece ţinînd seama de forma unui term avem evident T_k *(Ek ... ..., E„)* a T_k*(E[,..., E'„)* ■

Teorema 11.7.2. *Fie* **D<*+D =** (D<*+,> ,..., $D^{(}_{n}{}^{A+1}$>) **= H(D(»)),** Dl'> = *(D^>,...* **..., D(°') =** H(0,..., 0), *unde* **H** *este o funcţie vectorială standard. Atunci*

$$\text{(II.7.3)} \qquad \mathrm{D} = \underset{A=0}{(\mathrm{JD}'»)},$$

este o soluţie a sistemului (II.7.2) [11]

Demonstraţie. Vom verifica, prin inducţie că **1)1h~*) £ D<*) ,** V *li* A', unde Dl-'l = (0,...,0). Avem, evident pentru *h*= 0, (0,....0) £ D°. Apoi, să presupunem că **D(*—') £** Dl*) pentru *k* sg *h.* Rezultă :

$$\mathrm{D}('+i) \mathbf{= H(I)^{(/|)})} 3\ \mathrm{H(I)l''\text{-}D} = \mathrm{Dl*}),$$

conform teoremei II.7.1.

Mai mult, observăm că'pentru $k_L < k_2$ avem **D<*>) £** D(*«).

Fie deci

$$D, = \mathrm{U}\ ,$$

şi să calculăm a J-a componentă din **H (D):**

$$77,(\mathrm{D}) = \underset{\text{A-0}}{H,}\ (\mathrm{U}\ \mathbf{D<*))} = \underset{\text{A,-0}}{77,}\ ((\mathrm{J} \qquad \underset{\text{A„-0}}{} ,..., (\mathrm{J} \qquad =$$

U *H, (D*'>* ,..., 7)1*»)) = U 77, (D<»> ,..., D<?>) = (A,^,....^$_n$) e.v"

$$= (\underset{A\text{-}0}{J}\ D<*+D = \underset{A=1}{Q}\ D\hat{} = \underset{A\ —0}{Q}\ Dp = D_{\}}\ .$$

Am folosit aici faptul că H_f (D($_1^{A,}$)..... *D^n')* c $H_{\}}$ *(D'$_1^{h>}$,..., D^}),* unde h = max {/ij,..., h_n}. ■

Definiţia II.7.3. *Limbajele de tip* ALGOL[II'] sînt componentele soluţiei (II.7.3) a sistemului (II.7.2).

Teorema II.7.3. *Limbajele de tip* ALGOL *sini limbaje de tip* 2.

Demonstrafie. Fie **E = H (E) =** *(H$_k$ (E$_k$,..., E„),..., H„ (E$_l$,...,* E„)) un sistem în care H este o funcţie vectorială standard şi *Hj (E$_k$*.....E„) = "O)
= tj ,..., *E$_n$)* exprimarea componentelor lui H ca reuniune de fermi.

Astfel T_{jk} £ *K*,* unde *K = I* u {*E$_1$* ,..., E„} [III]>. Avem T_{jk} *(E) = t$_{jkl}$ (E)...* ••• Gtmd,») (£)> unde m (j, A) este lungimea cuvîntului T_{jk} *£ K*.*

Să considerăm gramaticile de tip 2, $G_{\}}$ = (K, *I, E$_{\}}$; F)* unde *F* = -> -> | A = 1, R (fi), *h* = 1, n}. Vom arăta că *L (G$_{\}}$)* = , unde **D** = *= (D$_1$,..., D$_n$)* este soluţia (II.7.3) a sistemului **E = H (E).** Mai întîi vom dovedi că D_s £ L (G_3). Pentru aceasta este suficient să arătăm că $D^{l,h\}}$ *£ £ L* (G_3), VA £ *N.*

Procedăm prin inducţie asupi.i lui *h.* Pentru *h* = 0 fie *p* £ Dj$^{0)}$ = *= Hj* (0 ..... 0). Atunci *p* este un term al lui $H_{\}}$ (E), *p = £ I*.* Prin urmare *Ej -> p* este o regulă de generare din *F* şi *p* fiind alcătuit numai din simboli terminali, *p £ L (Gj), deci D^{I}^ £ L (Gj).*

Să presupunem că O^$^{ft)}$ £ *L (Gj)* şi să arătăm că *D^$^{+u}$ £ L (Gj).* Fie *p*]D^$^{+1)}$ = *Hj* **(D<ft>).** Există A £ *N* astfel îneît *p £ T$_{jk}$* **(D<*>)** adică *p £ £ t$_{jkl}$* **(D<*>) ...** *t$_{jk\ m\{jik)}$* **(D<*>).**

Putem scrie deci *p* = z$_3$... *i$_m$(jtk)* unde

dacă $t_{\}kv}$ *£ I*

<j_v *£* , daca *tj$_{kv}$* = Ev(r) •

Ţinînd seama de ipoteza inductivă, 0% £ *L* ($G_{„(r)}$), avem, în

---

II*) **Denumirea engleză pentru aceste limbaje este ..ALGOL-Jike languages".**

III) **Reamintim că, pentru simplitate, am convenit să notăm cu aceeaşi literă termul şi limbajul generat de el.**

cazul z„= *q„* deducţia $E_{v(r)}$ $Z>i_v$,

Urmează deci că :

$$Ej\ hkl\ (E)\ \bullet\bullet\bullet\ hkmd\ k)\ (E) =^{*}\blacksquare\ ^{r}l\ \bullet\bullet\bullet\ Gj$$

adică *Ej => p* sau *p £ L (Gj).* Astfel D^+1) £ *L (Gj),* ¥*j* şi în definitiv *D*ȷ *L (Gj).*

Invers, să arătăm că *L (Gj) £ D*ȷ , Vj.

Să observăm, pentru aceasta că va fi suficient să dovedim că dacă r_k **(E),** r_2 **(E)** e *K** şi *i*| **(E) =>** r_2 **(E)** atunci r_k **(D) £** r_2 **(D).** Aceasta, deoarece dacă *p* s *L (Gj)* avem $E_ȷ$ => *p* si *Dj = H*ȷ (D) 2 *p* (I)) = {/;}, deoarece *p* fiind un ^{G}i

cuvînt din alfabetul *I* el nu depinde efccliv de I). Astfel *p* **e** *I)*ȷ şi deci *D (GJ* **S** *Dj .*

Fie deci *i*| **(E),** r_2 **(E)** **e** A'* şi r_k **(E) =>** r_2 **(E).** Să presupunem că în gramatica *Gj* avem deducţia : ^{a}j

'1 (E) = ^{u}o (^{E}) =" ^{u}i (^{E}) =>•••=> u» (E) = r_2 **(E).**

Pentru a arăta că ij **(D)** 2 r_2 **(D)** vom proceda prin inducţie după *h.* Fie, mai întîi *h* = 1. Atunci r_k **(E)** => r_2 **(E)** şi deci

G **(E) =** u_k **(E)** v_3 **(E)** v_2 **(E),** r_2 **(E) =** v_k **(E)** P_4 **(E)** v_2 **(E)** şi

(u_3 -> u_4) s *F.* Din definiţia lui *F,* există *j, k N* atsfel îneît u_3 **(E) =** $E_ȷ$, = T_{ir} **(E).** Cum *Dj* este soluţia sistemului (II.7.2) avem

Dj = Hj **(D) = 7',(D)** u U T_{jk} **(D).** Apoi P_3(D) = *Dj* = T_{jt} *(D)* u U (D) =*4=1 *4=1

= e_4 (D) U |J T_{jk} **(1))** 2 P_4 **(D).** De aici obţinem

'1 **(D) =** "1 (^{u}) *i*>$_3$ (B) v_2 **(D)** 2 v_k **(D)** e_4 **(D)** v_2 **(D) =** r_2 **(D).**

Presupunem că afirmaţia este adevărată pentru *h < k* şi fie *i*| **(E)** şi r_2 (E) două cuvinte astfel îneît :

$$^{1}1\ \textbf{(E) =}\ u_0\ \textbf{(E) =>}\ 11_{\ k}\ \textbf{(E) => ... =>}\ u_{t+1}\ \textbf{(E) =}\ r_2\ \textbf{(E).}$$

Din ipoteza inductivă avem mai întîi **(D)** 2 u_k (D) şi apoi u_k (O) 2 2 r_2(D), deci Zj(D) 2 r_2(D).B

Teorema II.7.4. *Limbajele de tip* 2 *sini limbaje de tip* ALGOL.

Demonstraţie. Fie *G = (I*$_N$ *, I*$_T$ *, x*$_0$ *; F)* o gramatică de tip 2. Fără a res- trînge generalitatea putem presupune că I_N — U {X_o T_s} deoarece adăugarea simbolilor terminali la cei neterminali nu influenţează limbajul n
generat de gramatică. Avem *F* = JF$_y$, unde F, = *{Tj -> r*$_k$*'> | k =*

1, T (j)}.

Să considerăm funcţiile standard H_j (x) = tj {rj/^{1}} unde x = (x_0, x_k,..., x_n). *-1 T(J) Gramtica G_o construită ca în teorema precedentă coincide cu G ; ca atare limbajul L (G) va fi prima componentă a soluţiei de tip (II.7.3) a sistemului **x = H (x)** unde **H =** *(H_0,...,H_n).*

Conchidem că L (G) este un limbaj de tip ALGOL. ■

Observaţia II.7.1. Fie E un limbaj de tip 2 generat de gramatica G = *(I_N, 1* F) şi F = tj *Fj* cu $F_{;}$. = {x_y -> rj/' I ^{k} = 1> T(J)} = U ^{F}*a* U U $^{F}_{ir}$, *i-o teij xeiff* unde *Fj$_t$ = [Xj -» ir / r* s J*}, $^{F}_{iX}$ = -» *xr / r e* /*}, Vom face presupu
nerea că dacă (x -> e) F atunci x = x_0 , fără a micşora generalitatea.

în acest caz sistemul de ecuaţie **x = H (x)** din demonstraţia teoremei precedente devine :

•Tj = U {')A} u U >ih u Zj ,j = 0, 71, *z^e/ rh^IN
0. clacă / / O sau j = O şi c E {e}, dacă j = O şi e e E

Teorema II.7.5. *Clasa limbajelor de lip* ALGOL *coincide cu clasa limbajelor de lip* 2.

Demonstraţie. Rezultă din teoremele II.7.3 şi II.7.4. ■

Exemplul II. 1.1. Fie I_T = {i\ , i_2, i_3} şi *II* (,r_0 , x_1) = ({i_x ,T_0) U {.TJ R i_2 .r_0} u U {i_3}, u {ij.ro} U {i_x i_3}).
Să considerăm gramatica G = ({.r_0 , .T_2) u {ij, i_s, i_3}, {'i , E > '$_3$}> r_0 '> *F)* unde

$$F = x_0 \to i_1 x_0 ,\ x_0 \to x_1 i_1 i_2 x_0 , x_0 \to i_3 ,\ x_1 \to x_0 x_1 , x_1 \to i_1 x_0 ,\ x_1 \to i_1 i_3 \}.$$

Atunci *L (G)* este prima componentă a soluţiei sistemului :

(II.7.4 $Eg = \hat{i}_t\ Eg$ U *l''l ij* i_2 *Eg* u {i_3}
1^1 = E_g E_t u ij *Eg* u {i\ i_3}.

Soluţia (11.7.3) a sistemului (II.7.4) poate fi construită astfel :

D<°) = 11(0,0) = ({i_3}, {ij i_3}),
I)(D = H(I)«») = ({,; j_3,$i_3 1_1$ i_2 i_3, ,-$_3$i,
. {'3 '1 >2 < '1 '$_3$})>

Şi I) = U $^{I)(n}$ •
$i=o$

Teorema II.7.G. *Soluţia* (II.7.3) *a sislemului* (II.7.2) *esle primul element al mulţimii soluţiilor sislemului.*

Demonstraţie. Conform teoremei II.7.2 mulţimea soluţiilor sistemului (II.7.2) este nevidă. Fie E = (Ej,..., E_n) o soluţie oarecare şi I) = (D_t,..., D_n) soluţia de tip (II.7.3).

Vom demonstra prin inducţie după *j,* că D<0 £ E, V) e N, de unde efi
va rezulta că **D** = (J D(E £ **E .**

Fie *j* = 0. Deoarece 0 £ **E** urmează **D<°) = H** (0) £ **H (E) = E** (v. teorema (II.7.1)). Să presupunem că D<0 £ E. Avem

D<J+1> = (D<0) £H (E) = E. ■

§.11.8. AUTOMATE PUSHDOWN (AUTOMATE CU PILE DE MEMORIE)

Ne vom ocupa în acest paragraf de o clasă de dispozitive care joacă pentru limbajele[1] de tip 2 acelaşi rol pe care-1 joacă automatele finite faţă de limbajele de lip 3, şi anume de

automatele pushdown.

Spre deosebire de automatele finite, *automatele push- doum* posedă atît memorie internă (a cărei activitate se exprimă prin evoluţia stărilor interne ale, automatului) cît şi o memorie externă de un tip special, memoria pushdown.

*O memorie de tip push- down*r> este o bandă nemărginită în jos alcătuită din celule în care se poate memora cîte un simbol dintr-un

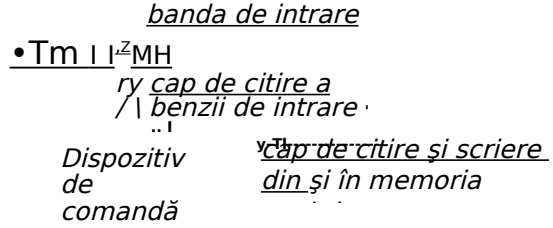


Fig. II.

introducind un simbol în prima conţinutul tuturor celorlalte se deplasează în jos cu o celulă, iar extrăgînd conţinutul primei celule se produce o deplasare în sus cu o celulă. Prin urmare. ultimul simbol introdus este in mod necesar primul simbol extras şi primul simbol introdus este ultimul extras.

Un *automat pushdown* lucrează in mod discret, în tacte şi conţine trei părţi :

— O bandă de intrare împărţită în celule care pot conţine cîte un simbol dintr-un alfabet de intrare *I,* numărul celulelor ocupate fiind finit.

— Un dispozitiv de comandă, care conţine memoria internă, capabil să se afle într-o anume stare dintr-o mulţime finită de stări S. Aceasta posedă un „cap de citire" a benzii de intrare şi un alt cap pentru citirea şi scrierea informaţiilor din şi în memoria pushdown.

— O memorie pushdown care conţine simboli dintr-un alfabet Z (v. fig. 11.8.1).

Funcţionarea unui astfel de automat intr-un anume tact depinde de simbolul citit de pe banda de intrare, de starea internă a dispozitivului de comandă şi de conţinutul primei celule din memoria pushdown.

Banda de intrare se poate deplasa în fiecare tact cu cîte o celulă la dreapta Irecînd prin faţa capului de citire sau poate rămîne pe loc în care caz se consideră că dispozitivul citeşte cuvîntul nul. în citirea unui simbol de intrare din 7tJ {e} dispozitivul de comandă efectuează două acţiuni funcţie de sta-

rea în care se află, de simbolul de intrare citit şi de conţinutul primei celule a memoriei pushdown :

— trece într-o nouă stare internă

— înscrie în memoria pushdown înlocuind primul simbol cu un cuvînt / Z^{IV} (care poate fi eventual cuvîntul nul, în care caz se produce de fapt ştergerea conţinutului primei celule). Bineînţeles că la introducerea unui cuvînt de lungime $k > 0$ se produce deplasarea în jos a conţinutului memoriei pushdown cu k celule.

începerea activităţii automatului se face dintr-o stare, starea iniţială s_0 şi un anumit conţinut "$_0$ eZ al primei celule din memoria pushdown numit *simbol de start.*

Recunoaşterea unui cuvînt de pe banda de intrare are loc cînd, după citirea ultimului său simbol memoria pushdown s-a golit complet^ Formalizarea acestei noţiuni se face în :

Definiţia II.8.1. Un *automat pushdoivn* este un 6-uplu *APD = (I,S,Z, f',s₀,* z_0) unde 7, S, *Z* sînt mulţimi finite nevide reprezentînd respectiv alfabetul de intrare, mulţimea de stări interne şi alfabetul memoriei pushdown al automatului, *f:S* x (7 tj {e}) x Z -» *(p_f (S x Z*)* numită funcţia de tranziţie, *s_0 e S* este starea internă iniţială, iar z_0 e Z simbolul de start.

Prin *stare globală* a automatului pushdown înţelegem un cuplu (s, *q)* e S x Z*.

Observaţia II.8.1. Automatul pushdown descris în definiţia II.8.1 funcţionează în mod *nedeterminist* în sensul că în fiecare tact el are posibilitatea mai multor alegeri pentru noua stare internă şi pentru cuvîntul înscris în memoria pushdown. Avem totdeauna ! *f* (s, *i,* z) | < oo , iar dacă | *f* (s, *i,* z) | = = 1, *Vs* s S, *i* s 7 u {e}, z s Z automatul funcţionează în mod *determinist.* Dacă *f (s, i,* z) = 0 atunci simbolul *i* este inaplicabil la intrarea au. tomatului cînd starea internă este s şi z se află în capul memoriei pushdown.

Definiţia II.8.2. O *configuraţie instantanee* a unui automat pushdown *APD* = (7, *S, Z, f;s₀ , x₀)* este un triplet *8 = (s, p, q) S x I* x Z*.*

Definiţia II.8.3. *Configuraţia instantanee* 3_X = (s, *ip, zq),* unde *i* e *I* (J |e], z s *Z* precede imediat configuraţia instantanee S_2 = ($', *p, q_1 q)* dacă şi numai dacă ($', <7J) G *f(s,i,* z), ceea ce se va nota cu 3j |—S_2.

Configuraţia instantanee 8 precede configuraţia instantanee 8', dacă există un şir finit de configuraţii instantanee 3 = 8_0 , 8] ,..., 3„ = 8' cu 8_f /— 8_{f+1}, *j* = 0, n — 1.

IV **Denumirea de „pushdown[1]* descrie sugestiv funcţionarea memoriei. Ea provine de la cuvintele englezeşti to pusli = a Împinge şi down — jos.**

*

Vom nota aceasta cu 3 |— 3'. Şirul 3 = 3_0 |— $8_{\pm}$ |— ... |— 8_n = 8' îl vom numi *calcul al configuraţiei 8'* pornind de la configuraţia 8, iar *n* va fi numit *lungimea calculului.*

Observaţia II.8.2. Configuraţia instantanee (s, *p, e)* nu poate precede nici o altă configuraţie instantanee deoarece funcţia *f* nu este definită pentru triplete de forma (s, *p,* c). Din acest motiv, la începutul activităţii în memoria pushdown se află totdeauna simbolul de start z_0 £ Z.

Observaţia II.8.3. Configuraţia instantanee *(s, p, q)* reprezintă faptul că automatul se găseşte în starea internă *s,* că în tactele următoare se va prelucra cuvîntul *p* de pe banda de intrare şi că în memoria pushdown se află cuvîntul *q* scris de jos în sus, simbolul c«l mai de Ia dreapta al cuvîntului aflîndu-se în prima celulă (din ,,vîrf“).

Relaţia |— S_2 indică faptul că automatul se află în starea *s,* că z este simbolul din „vîrful“ memoriei pushdown şi că *i* este simbolul de intrare care trebuie să fie prelucrat sau că *i* = e şi singura evoluţie posibilă prelu- crîndu-I pe *i* este trecerea în starea *s'* şi înlocuirea lui z prin cuvîntul q_r.

Relaţia (5, i_k *p, q)* H ($', *p, q')* indică faptul că automatul se află în starea s, *q* este conţinutul memoriei pushdown, că *... i_t p* este partea ce urmează să fie prelucrată la intrare şi după *cel puţin k* tacte (deoarece în unele tacte banda de intrare nu se deplasează) automatul trece în starea *s',* conţinutul memoriei pushdown este *q'* şi rămîne de prelucrat cuvîntul *p* de pe banda de intrare.

Automatele pushdown pot servi la recunoaşterea unor limbaje aşa cum rezultă din :

Definiţia II.8.4. Limbajul *L (APD)* recunoscut de automatul *APD* =
: (/, 5, Z_t *f y* S_Q *y* ZQ) este

L (APD) = {p | 3s e S, (5^ , *p,* z_0) P- (5, *e,* e)}.

Vom ilustra noţiunea de automat pushdown în :

Exemplul II.8.1. Fie *APD* = ({ij , i_2}, {s_0}, {i_x, r_2, z_0}, *f; s_n ;* z_0) unde *f* este dată de următorul tablou :

f(s_a, i, z)	h		*zo*
'1	{('«o, «)} 0	0 {(%. «>}	0 0

e	0	0	R^{s}o > G^{x}(>y» h'2)}

Introducînd cuvîntul $p \bullet = i_r\ i_k\ i_t\ i_2\ i_2$ '2 putem obţine, în funcţie de alegerile pe care le face automatul mai multe şiruri de precedenţe imediate printre care avem :

(^{S}O > ^{l}l *1 *1 *2 *2 *2' ^{Z}o) I(^{S}0 ' *1 *1 *1 $^{(}$2 *2 ^{l}2 ' ^{l}l ^{Z}O '2) I

I(^{S}0 ' *1 $^{!}$1 *2 *2 *2 ' ^{Z}0 *2) I (^{S}0 > *1 ^{f}2 *2 $^{!}$2 ' ^{l}'l ^{Z}O *2 ^{h}\) I

I— (^{S}0 > ^{l}2 ^{l}2 ^{f} 2 ' ^{Z}0 *2 ^{l}2) I— (^{S}O ' ^{l}l ^{r}2 ^{l}2 *2 > L *2 ^{l}2 G) I—

I— (^{S}0> ^2 ^2 ^2 ' ^2 *2 *2) I ('**<) '^2 '2 > G ^2) I («O > ^2 > *2) I («O '

sau, de pildă, şirul :

(•"'o - «i '1 '1 '2 '2 '2 , ^{z}o) H (^{s}o > '1 '1 *h* '2 «2 > *'1* '2) H

I— (^{S}0 ' *1 ^{f} l ^{l}2 $^{(}$2 G > *^2)'*

care nu mai poale continua simbolul ij nefiind aplicabil atunci cînd automatul se găseşte în starea s_0 avînd în capătul memoriei pushdown simbolul i_2 . Cu toate acestea, existenţa primului şir de precedenţe imediate, adică

(^{S}0 > G '1 *n* '2 *2 *2 ' 'o) I (^{S}0 • $^{e'}$

asigură că ij $i_2\ i_2 i_2\ ^{s}$ *L (APD).*

Se verifică imediat că :

$$L\ (APD) = \{ij... \text{ ij } i_2... \ i_2 \ / \ n \text{ e } N^*\}.$$

Pe lîngă definiţia II.8.4 a limbajelor recunoscute de automate pushdown (în care recunoaşterea unui cuvînt se face prin golirea memoriei pushdown) vom prezenta un alt punct de vedere apropiat de reprezentarea limbajelor regulate în automate finite descris în :

Definiţia II.8.5. *Limbajul reprezentat intr-un automat pushdown APD = = (I, S, Z, f; s_0 , z_0) prin mulţimea de stări finale* Sj c S este :

$$L\ (APD)\ (SJ = \{p\ 3<7\ G\ Z^*.\ s\ e\ S_{lt}\ (s_0\ ,\ p,\ z_0)\ H\ (s,\ e,\ q)\}.$$

Legătura dintre definiţiile II.8.4 şi II.8.5 este dată în :

Teorema II.8.1. *Un limbaj L este recunoscut de un automat pushdown dacă şi numai dacă este reprezentat intr-un automat pushdown de o mulţime de stări finale.*

Demonstraţie. Fie *APD = (/, S, Z. f*; s_0, z_0) un automat

pushdown, mulţimea de stări finale S_J G S şi

L *(APD)* (SJ = {p| 3<7 s *Z*. s* s S_p (s_0, *p. z_0) /— (s. e, q)}.*

Să construim un automat pushdown în care acest limbaj să fie recunoscut. Pentru aceasta fie :

$$A' = (1, S \text{ u } \{«', \hat{}0\}' \text{ z U } \{z'\}, f' ; s'_o, z').$$

unde funcţia *f* este definită prin :

$$(s, i, z) \quad \begin{array}{l} (s_0, z_0, z^f), \text{ dacă } (s, i, z) = (s_0. c, z') \\ (s', e), Vs \text{ s Sj u } \{s'\} > i = e, z \; Z \; u \; \{z'\} \\ f(s, i, z), \text{ V.s G S, } i \text{ s } 1 \text{ u } \{e\}, z \text{ e } Z \; 0, \text{ în orice alte situaţii.} \end{array}$$

Introducerea simbolului *z'* pentru automatul *A'* se justifică prin faptul că pot exista cuvinte în *I* | L (APD)* (SJ care să realizeze în automatul *APD* golirea memoriei pushdown. Acest lucru nu mai este posibil în automatul *A'* din cauză că simbolul *z'* este prezent în ultima celulă a memoriei pushdown. Egalitatea *f'(s'o, e, z') = (s_Q, z_0 z')* exprimă faptul că automatul *A'* îşi începe funcţionarea prin trecerea din starea sj, în starea s_0 şi prin plasarea cuvîntului z_0 z' în memoria pushdown. Dacă *APD* intră într-o stare finală sub influenţa unui cuvînt *p,* sub influenţa aceluiaşi cuvînt automatul *A'* poate trece în starea *s'* şi apoi să efectueze ştergerea memoriei pushdown inclusiv a simbolului z'.

Fie *p* s *L (APD)* (SJ. Atunci există un calcul

$$(^{s}o> P. \, ^{z}o) \; H \underset{APD}{} (^{s}> \, ^{e}> ?)> \, ^{4\,e\,s}i-$$

pe care-1 folosim pentru a pune în evidenţă un calcul în automatul A'care să conducă la golirea memoriei pushdown.

în baza definiţiei funcţiei *f'* avem:

$$(«i, P. z') \; I- \underset{A'}{(^{s}o> P, Zo \; z')} \underset{A'}{|-} (s, e, q \; z') \underset{A'}{} H (s', e, e).$$

Astfel *p* s *L* (A').

Invers, fie *p £ L (A').* Există un calcul

$$(4 \; P> z') \; H \underset{A'}{} (s, e, e). \qquad \text{(II.8.1)}$$

Prima configuraţie în care intră automatul *A'* este în mod necesar (s_0, *p, z_0* , z'). Astfel, calculul (II.8.1) se mai scrie

(4 P, z')H (s_0, *P,* z_0 z') H (s, e, *e).*
A'

Din definiţia funcţiei *f* rezultă că configuraţiile care se succed sînt cele ale automatului *APD* cu excepţia prezenţei simbolului z' din celula cea mai de jos a memoriei pushdown. După prelucrarea cuvîntului *p* configuraţia automatului *A'* va fi (s, e, *qz'),* unde *q £ Z* şi s e S.

Simbolul z' nu poate fi şters, aşa cum se remarcă din definiţia funcţiei *f'* decît plecînd de la o stare din Si sau din starea *s'.* Avînd s e *S,* automatul *APD* nu poate trece în starea *s'* şi cum ştergerea simbolului z' are loc în orice caz rezultă că automatul APD trebuie să treacă în mod obligatoriu într-o stare finală (din mulţimea S_t) sub influenţa cuvîntului *p.* Aşadar s £ Sj şi avem (s_0, *p,* z_0) |— (s, e, 7), adică *p £ L (APD)* (SJ. Astfel *L* (A') = ***APD*** *= L (APD)* (SJ.

Reciproc, fie *APD = (I, S, Z, f;* .s_0 , z_0) un automat pushdown şi *L (APD)* limbajul recunoscut de acest automat. Vom construi un automat pushdown în care acest limbaj să fie reprezentat printr-o mulţime de stări finale. Fie deci A' = *(I, S* u % U *f;* -'o. z_0*)* un automat pushdown în care *f* este dat prin :

$f\,(s_0, z_0) =$ (s_0, z_0 Z'X dacă (s> z) = (4 *e,* z') *f (s, i, z),* Vs e S, i e 7 (j {e}, - s / *(s_f , e),* Vs s S, *i = e, z* = z' *0,* în orice alte situaţii.

Vom arăta că *L (APD) = L (A') ([s,}),* adică în automalul A' limbajul *L (APD)* este reprezentat prin starea finală {.$,}.

Fie *p* e *L (APD).* Există un calcul :

$$(\text{«o . P. z o}) \underset{APD}{\vdash} (\text{s- e> <0-}$$

Să considerăm configuraţia iniţială (s,, *p,* z') în automatul A'. Configuraţia care o succede imediat pe aceasta nu poate fi decît (s_0 , *p,* z_0 z') aşa cum rezultă din definiţia funcţiei *f.* în continuare configuraţiile care se succed sînt cele ale automatului *APD* cu excepţia prezenţei simbolului z' în celula cea mai de jos a memoriei pushdown a automatului A'. Atunci, după prelucrarea cuvîntului *p* putem obţine în A' configuraţia (s, *e, z')* şi automatul A' poate trece în configuraţia *(s_f, e,* e), deci *p* e *L* (A') ({s_z}).

Invers, fie *p* e L(A')({s_z}). Există un calcul

(I1-8.2) *(s_0, p,* z') H *(s_f, e, e),*

deoarece automatul A' nu poate ajunge în starea s_t decît la ştergerea ultimului simbol *z'* din memorie.

Calculul (II.8.2) poate fi detaliat :

$$(s_0', p, z') \underset{A'}{\vdash} (s_0, p, z_0, z') \overset{*}{\underset{A'}{\vdash}} (s, e, z') \underset{A'}{\vdash} (s_f, e, e).$$

Corespunzător calculului (s_0, p, z_0 z') |— (s, e, z') în automatul *APD* avem *A'* calculul

$$(\text{s o> P, z o}) \underset{APD}{\vdash} (\text{s> e> e})'$$

deci *p* s *L (APD).* ■

în continuare vom pune în evidenţă legătura dintre limbajele de tip 2 şi limbajele recunoscute de automatele pushdown. Mai precis, vom arăta coincidenţa celor două clase de limbaje.

Teorema II.8.2. *Dacă L este un limbaj de tip* 2 *există un automat pushdown in care limbajul L poate fi reprezentat.*

Demonstraţie. Să presupunem mai întîi că *e L.* Fie *G = (I_N, I_T, x_Q ; F)* o gramatică de tip 2, de forma normală S. Greibach (v. teorema II.5.12) care generează limbajul *L.* Fie A = *(I_T,* {s}, *I_N,f;s, x_0),* unde *f(s,i,x)* = = (s, r), dacă *(x -> ir) e F* şi *r* s *I'$_N$* şi arbitrar în rest. Vom arăta că *L = = L* (A).

Fie *p* = q... i_n *& L.* Există deducţia x_0 =*• $p_{,G}$* care se poate scrie detaliat: $\underline{x_{0,} \Rightarrow}$ i_t q => i_x i_2 r_2 => ...=> ... i_{n_1} $r_{B_}$j => i_t... $i_{„}$ = p'),

unde r, G r_N ,
j = 1, n — 1 şi poate fi considerată, fără restrîngerea generalităţii ca deducţie imediată la stînga (v. teorema II.5.6).

Fie (s, p, x_0) configuraţia iniţială a automatului A. Conform definiţiei funcţiei f în automatul A putem avea calculul :

$$(s, ij\ldots i_{„}, x_0) \text{ j}\underset{A}{—} (s, i_2 \ldots i_{„}, \text{ rj} \underset{A}{|—} (s, i_3 \ldots i_{„}, r_2) \underset{A}{(—} \ldots$$
$$\ldots \text{H } (s, i_{„}, \underset{A}{|—} (s, e, \text{e}),$$

deci $i_2 \ldots i_n = p$ e L (A).

Invers, fie p e L (A). Există calculul

$$(s, p, x_0) \text{ H } (s, e, \text{e}).$$

Vom arăta, prin inducţie asupra lungimii cuvîntului p că dacă există

• •

un calcul (s, p, r) $\underset{A>}{|—}$ (fi, e, e) atunci există o deducţie $r \underset{G}{=>} p$.

Fie l (p) = 1, adică p = i G I_T . Avem (s, i, x_0) $\underset{A}{|—}$ (s, e, e) şi corespun-

zător avem regula de generare $(x_0 \text{ ->} i\ )$ e F şi deci i G L.

Presupunem afirmaţia adevărată pentru cuvinte p cu l (p) < n — 1 şi fie q G I^*_T cu l $(q) = n$ şi $q = ip$, cu l (p) = n — 1.

Deoarece (.s, q, x_0) = (s, ip, x_0) j— (s, p, r) j— (s, e, e), unde $(x_0$ -» $ir)$ G F, în A A^* baza ipotezei inductive avem deducţia r => p, de unde rezultă existenţa deducţiei $x_0 => ir => ip = q$, deci $L = L$ (A).

Dacă e e L considerăm limbajul {e}, ataşăm gramaticii sub forma normală S. Greibach regula de generare x_0 -> e şi admitem că f (s, e, x.) = = (s, e), adică permitem automatului A să prelucreze ţi cuvîntul nul cfec- tuînd ştergerea memoriei pushdown. ■

Teorema II.8.3. *Dacă limbajul L esle reprezentat într-un automat pushdown, L este un limbaj de tip* 2.

Demonstraţie. Fie $APD = (I, S, Z, f; s_0 , z_0)$ un automat pushdown şi $L = L$ (APD). Vom construi o gramatică de tip 2, G astfel ca $L = L$ (G). Anume, $G = (I_N , I, x_0 ; F)$, unde 1_N = {s_0} |J {(s_x, z, s_2)' s_x, s_2 G S, Z <= Z}, iar F este mulţimea regulilor de generare care se pot forma prin următoarele două procedee:

1) x_0 -> (s_0 , z_0 , s), Vs e S

2) (s, Z, S) -* i (s_l , Zj , S,) (^{s}2 > “2 > $3) "• ('m > ^2m ’ $^S m+1$) pentru Vs, Sj ,... .... ^{s}m+i £ S cu $s' = s_{m+1}$, VIG / J $\{e\}$, z, z_x,..., z_m G Z, astfel ca (Sj, z_t, z_2,..., z_m) G f (s, i, z). Dacă m = 0 atunci Sj = s', (s', e) f (s, i, z) şi regula de generare devine (s, z, s') -» i, unde i G J ţj {e}.

***> Remarcăm că ultima regulă de generare Întrebuinţată este In mod necesar de forma**

Să presupunem că *p & L(APD),* adică există calculul

(11.8.3) $(s_0 , p, z_0) \underset{APD}{H} (s, e, e).$

Vom arăta că $x_0 \underset{a}{\bullet}\Rightarrow p$, adică *p L (G).*

Pentru aceasta, vom demonstra prin inducţie după lungimea calculului, că dacă există calculul :

(11.8.4) $(s, p, z) \underset{APD}{H} (s, e, e),$

atunci există deducţiā $(s, z, s) \underset{o}{\bullet}\Rightarrow p.$

Fie un calcul (II.8.4) de lungime n = 1. Atunci *l (p)* 1 şi *p = i* e /(J {e}.
Astfel, (s, i, *z) /— (s, e, e), (s, e)* ^{E} *f (s, i,* z) şi prin urmare *((s, z, s) -> i) <s F* şi deci există deducţia (s, z, s) => i.

Să presupunem că afirmaţia este valabilă pentru calcule (II.8.4) de lun-

gime cel mult *n* — 1 şi fie (s, *q,* z) |— (s, *e, e)* un calcul de lungime n. Avem *APD*
l (</) < n şi *q = ip,* cu *l* (p) < n — 1. Calculul de mai sus poate fi scris şi :

(s, *q,* z) = (s, *ip, z)* H (St, *p,* r) = $(s_x , p_L \ldots p_m . z_1 \ldots z_{„})$ |— (s, *e, e),* cu ni > 1

astfel încît Vj = 1, *m* are loc calculul

$$(,Sj , p_t , z,)\underset{APD}{H} (s,_{+1} , e, e).$$

avînd lungimea cel mult n — 1. Aici s_{m+1} = s.

Cuvîntul *p* a fost împărţit în subcuvintele p_t. $p_2,\ldots, p_m$ astfel încît în urma prelucrării cuvîntului p_j să se realizeze ştergerea simbolului z_t din memoria pushdown, *j* = 1, *m.* în acest caz, conform ipotezei inductive există deducţiile :

$$(*/ - \ > {}^{s}/+i) \underset{a}{\Rightarrow} fb \bullet .7 = 1, m, s_{m+1} = 5.$$

Cum in gramatica *G* avem regula de generare :

$$(s, z,s)^i (s, z_t , s_2) (s_2 , z_2, s_3)\dots (s_m . :_m, s_{m+1}).$$

(deoarece $f(s, i, z)_3$ ($j , z, , z_2\dots z_m$)) rezultă că :

$$(s, z, s) 4\text{-} i Pi \dots p_m = q.$$

Dacă *p* e *L(APD)* atunci există calculul:

$$('o \bullet P\text{-}^z o) \underset{APD}{H} (^?. \quad ^e)\bullet$$

de unde, in gramatica *G* avem deducţia.;

$$io \underset{o}{=}* («o - ^z o > P) => \underset{a}{P'}$$

adică *p & L* (G).

Să arătăm acum că *L* (G) £ *L (APD).* în acest scop vom demonstra că dacă are loc deducţia imediată Ia stînga :

(11.8.5) $' (s_0, :_0, s) ip (s\text{«} , z_a , sp)\dots (s_u, z_u, s),$

atunci există calculul :

$$(s\# \blacksquare P> -"o)î\text{-} (^s\text{-} ^e> ^z\text{«} -^z u)' \quad (APD.)$$

procedînd prin inducţie asupra lungimii *li* a deducţiei (II.8.5). Aici *p* e /». Pentru *h* = 1 cuvîntul *p* este obligatoriu de forma *p = i* e f (J {e}, deducţia (II.8.5) revine la :

$$(^5 0. ' *0 \quad ^1 \quad » ^z\text{«} > ^5 ă) \quad > ^z U ' ^s)'$$

şi (s_a , z_a ... 'u)e *f (s₀ , i,* z_0). Avem corespunzător calculul :

$$(*o\ ^z o)\ H\ (*\text{-}\ ^e>\ ^z\text{«} \bullet\bullet\bullet\blacksquare\ ^z u)\text{-} \quad (APD)$$

Presupunem afirmaţia valabilă pentru $h < n$ — 1 şi fie (s_0 . $:_0$,5) $\underset{G}{=>}$ *q* (s_v , s_y) ... (s^, z_u , .s) o deducţie de lungime n. Această deducţie poate *a* fi detaliată sub forma :

$$(\$o , *0 > \bullet^{<i}) \underset{G}{} P (^5 a , -Ol ? Sp) \dots (s^ , \quad , s)$$

$$=> pi (s_v, , Sy) \dots (s_6 , I_S , S_0) \dots (s_u . Z^*, 5).$$

unde *q = pi,* ultima regulă de generare folosită fiind G^ , . s_0) -> *i* (s_v. ,s_Y) ...

• •• G^B » şi deci *f* (s_a , *i,* z_a) s (s_v, ... x§).

Pentru cuvîntul *p* există deci deducţia :

(ˢ0 ’ *0 ’ P) *P* ($«» *« ’ Sp) ••• ($n , ‘ţi

de lungime cel mult *n* — 1 şi, conform ipotezei inductive exista calculat:

(ˢo - P- ᶻo) H (ˢo *,e,zₐ... 2ᵤ).*

Ca atare, plecînd de la configuraţia (s_n , *q. z_0)* = (s_0 , *pi,* r_0) obţinem calculul :

(ˢ0 ’ P'> ᶻo) H (■ˢ« . '• ᶻ« ••• -’u) H (ˢV • «> '« ••• ᶻu),

ceea ce. ne-am propus
să dovedim.

Fie acum *p L* (G). in baza teoremei II.5.6 există o deducţie imediată la stînga :

$$\bullet'o =^{*}_{G} (^{s}o - {}^{z}o - {}^{i-}) =^{*}_{O} P\,(^{5}« \blacksquare\ {}^{z}« - {}^{s}p) - \ .\ {}^{z}i^{*} \bullet {}^{s}) => P.$$

ultimele reguli de generare aplicate fiind (s_a , z_a , s_p) => e,..
„ (s^ , z^ , s) -* e.

Conform celor arătate mai sus există, pe de o parte un calcul :

(11.8.6) (s_0 , *p,* z_0) $\underset{APD}{H}$ (s. , *e, z_a ... z_u),*

iar pe de altă parte, din definiţia gramaticii *G* avem :

(fp . «) ᵉ f (s. . ₑ, z_a),

(s, *e) f (*$_{S/1,\ e}$*, z*)*.*

Astfel calculul (11.8.6) poate fi continuat, obţinîndu-se calculul :

$$(^{5}o - P> {}^{z}o)\ \underset{APD}{I—}\ (^{S}« \ ’\ {}^{e}> \bullet\bullet\bullet\ {}^{2}u)\ \underset{APD}{I—}\ (^{s}> {}^{e}<)’$$

ceea ce arată *că p L (APD).* ■

Teorema II.8.4. *Următoarele afirmaţii sini echivalente :*

1) *L este un limbaj de tip* 2.
2) *L este recunoscut de un automat pushdoum.*
3) *L este reprezentat intr-un automat pushdown printr-o mulţime de stări finale.*

Demonstrafie. Rezultă” din teoremele II.8.1 - II.8.3. ■

§11. 9. AUTOMATE DE TIP LETİCEVSKI

Vom prezenta în acest paragraf o clasă de automate deterministe infi- nite^care permit de asemenea recunoaşterea limbajelor de tip 2.

Această clasă de automate a fost introdusă de A. A. Leticevski în anul 1965 [1].

Definiţia II.9.1. *Un automat de tip Leticevski AL* este un 4-uplu :

$$AL = (Z,\hat{}(Z^*), F; L_o),$$

unde *I* este un alfabet finit, $<p_f$ (Z*) este mulţimea stărilor automatului *AL* (constituită din mulţimea limbajelor finite pe alfabetul finit Z) şi $F : <p_t$ (Z*) x xZ — este funcţia dc_o tranziţie a automatului *AL* care satisface
condiţiile :

1) F(0,i) = F({e},i) =0, Vi e *I,*
2) F (zp, i) = F (z, i) *p,* pentru Vi e Z, z e Z, *p* e Z*,
3) F (L_xU L_2 ,*i) = F(L_{lt}i)|J F (L_2*, i), Vi e *I,* L_t , L_2 e $(J>_f (Z^*)$,
4) Pentru *p* s *F* (z, i) avem *l* (p) < 2.

Observaţia II.9.1. în baza condiţiilor 1)—3) funcţia F este unic determinată prin specificarea mulţimii finite de limbaje {F(z, i)| z e Z, i e J},

Condiţia 2) este asemănătoare descrierii funcţionării memoriei pushdown, deoarece sub influenţa simbolului de intrare *i* este prelucrat numai simbolul z, începutul cuvîntului zp.

Exemplul II.9.1. Fie *AL* = ({i_x , i_2}, $<p_t (Z^*), F ; L_o)$ unde Z = {z_0 , zj, L_o = {z_0 , zj, iar F este determinat de tabelul :

F	*0	»i
'i	{*0. Mi)	{«1» Vo)
^8		{*0}

Observaţia II.9.2. Funcţia $F : rp_f(Z^*) \times I\hat{}><p_f(Z^*)$ poate fi extinsă la $<p_f(Z^*)$ x Z punînd :

$$F\ (L, e) = L_a\ VL\ s\ cp, (Z^*),$$

$$F\ (L,\ pi) = \mathrm{F}\ (L,\ p),\ i),\ Vp\ \mathrm{e}\ I^*,\ i\ \mathrm{e}\ I,\ \mathrm{VL}\ \mathrm{s}\ \hat{}>_z(\mathrm{Z}^*).$$

Definiţia II.9.2. *Limbajul LţAL) reprezentat în automatul AL* **este:**

$$L(AL) = \{\mathrm{p} \mid \mathrm{F}\ (\mathrm{s}_0\ ,\ \mathrm{p})\ \hat{}(\mathrm{Z}^*)\},$$

unde am notat prin $<p^e_f(Z^*)$ familia de limbaje finite asupra lui Z* care conţin cuvîntul nul *e.*

Vom pune în evidenţă o proprietate a limbajelor de tip 2 care ne va permite să arătăm coincidenţa acestei clase cu clasa limbajelor reprezentate în automate *AL.*

Definiţia II.9.3. O *sp-algebră* este o familie *CU* de limbaje asupra unui alfabet finit Z închisă la operaţiile de reuniune şi produs, adică *, L_2^ CU* implică LjU *L_2,$L_x L_2$<^ CU', sp-algebra CU,* (L) *generată de familia de limbaje L = {Lj|j* s J} este clasa limbajelor care se pot obţine prin aplicarea de un număr finit de ori a operaţiilor de reuniune şi produs asupra limbajelor L = {L,IJEJ}.

Dacă | *J* j < oo sp-algebra *CU (L)* se zice *finit generată.*

Definiţia II.9.4. O *sp-algebră CU* este *secvenţială* dacă VLe *CU* Şi < s Z, 8_t (L) 6 *CU,* unde

$$(L) = \{r \text{ j } ir \text{ e } L\}.$$

Aplicaţia *8_4 cp (/*) -> cp* (Z*) se numeşte *derivare la stingă in raport cu i* şi se observă că :

(II.9.1) $$\&i\ (\hat{}i\ U\ L_2) — 8_t\ (Z_n)\ (J\ 8_t\ (L_2),$$

ci

7^1

(II.9.2) $$L = U\ \hat{}LJ\ e(L), <6/$$

unde

$$_\ 1^{\ 01\ daCă\ e}\ \&\ ^{L}|\ e, \text{ dacă } e \text{ E } L,$$

iar 8_t *(L_x L_2) =* 8_t (L,) *L_2* (J e (LJ 8_t (L_2), ceea ce în cazul *e* L_t revine la

(11.9. $$(Aj\ L_2) —\ \ (-\hat{}î)\ \blacksquare$$

Observaţia II.9.3. Generatorii unei sp-algebre secvenţiale, finii generate, sînt în mod necesar limbaje de tip 2. într-adevăr, fie {L_t,..., *L_n}* un sistem de generatori ai algebrei *CU.* Avem 8_t *CU* şi deci $8_(\ L_k$ = (Lj ,..., *L_n),* oricare ar fi *i* E *I, k* = 1, *n* şi cum L_t = (J id_f *(L_k)* u e(L_t) obţinem

$$A_t = u \ldots\ ^{7}\text{-n)U}^{\ f}\ (\hat{}).^{\ k} = !\text{-}^{n'} \quad \text{«ei}$$
$$<er$$

unde <p_{ft} sînt funcţii standard.

Ecuaţiile de mai sus constituie un sistem de forma (11.7.2) şi cum sistemul are soluţie unică (în baza observaţiei 1.5.6), ţinînd seama de teorema II.7.3, componentele acestei soluţii sînt limbaje de tip 2.

Observaţia 11.9.4. Dacă sp-algebra *CU* este secvenţială atunci orice limbaj L_o *CU* este reprezentat în semiautomatul A= (Z, *CU, f, L_o)* prin mulţimea de stări {Z, | LE *CU,* e E L}, iar *f (L,*

i) = 8_f *(L), VLe CU, Vi* e Z.

într-adevăr, dacă *p e* L_o atunci *e* E *f* (L_o , *p)* deoarece $8_p(p) = e.$ iar dacă eE *f* (L_o , p) avem *p E* L_o .

Vom studia sistemul de ecuaţii :

(11.9.4) 7-J, = *F, (*E_x. E_n , *E,….*E_n*). J* = 1, n.

unde F_f *(Ej* E_n , L_n*), j* — 1, n sînt funcţii standard peste un al
fabet *I* (v. definiţia II.7.2).

Soluţia minimală a sistemului (11.9.4) este, conform teoremelor 11.7.2 şi 11.7.6

$$Dj = \underset{»-o}{U} .$$

unde *D?′ = F,* (D'*"[1]’ ...., DJ*-[1]’. DJ*-[11]..... iar

$$DJ^{01} = Fj\ (0. .0,\quad 0. .0).$$

Să observăm că această soluţie se. mai poate scrie :

(11.9.5) $$Dj = \underset{k-1}{tj}\ FJ^{*1}\ (0,.... 0,0,..., 0).$$

unde *F?′ = Fj* (FJ*-[11]……FJ*"[11], FJ*-[1]’..... FJ,*"[11]), iar

F/> = *Fj .*

Fie acum sistemul :

(11.9.6) *Lj = Fj* (JJ,… *t, , L.* ..... L„), *j* = Î77 şi ..... u n *1* = 0-

Să notăm $FJ^{m+11} = F_j$ *(^ ,..., £„ , F*|^{mJ},..., FJ,”*[1]), unde $FJ^{11} = Fj.$ Soluţia minimală a sistemului (II.9.6) este dată, în baza aceloraşi teoreme de relaţia :

$$Pj = tj\ />'*>$$

unde FJ*[1] = *Fj (£, … ,* F<*-”„... FJ,»-») şi

$$P? = Fj\ \ldots\ldots\ 0\ldots\ldots 0).$$

Această soluţie se mai poate scrie :

$$F, = \underset{k=l}{IJ}\ FJ^{*1}\ (^\ ,\ldots,\quad .\ 0..0)$$

cu F)*[+1]] = *Fj (^ ,..., l*$_n$, FJ*[1]. FJ.*[1]), iar $Fj^{11} = F_t$.

Să notăm cu *Q* limbajul care provine din limbajul *Q* c ({^ £,.

{tj..... ^)*
prin .‘uhstituirea simbolilor cu *Dj, j* = 1, n.

Au loc :

Teorema JI.9.1. *Avem :*

$$FJ^{ml}\ (?j\ldots E_{\prime\prime}\ ,\ 0,\ldots,\ 0) = F)^{ml}\ (D,\ldots D_{\prime\prime},\ 0,\ldots,\ 0).$$

Demonstraţie. Vom proceda prin inducţie după *m.*

Pentru *m* = 1 obţinem

$$F,\ «1\ \ldots\ Zn\ \blacksquare\ 0\ \text{-},\ 0) =\ (O.\ \ldots\ D_{\prime\prime}\ \ ,\ 0,\ldots,\ 0).$$

relaţie adevărată deoarece exprimă convenţia de notaţie făcută mai sus. Să presupunem acum relaţie adevărată pentru *m.*

Atunci avem

$$F\}\text{™}*”\ \ \ldots\ t_n\ .\ 0\ \ldots..\ 0) = F,\ \ t_n\ .\ FM\ \ \ \ \ \ .\ 0\ 0).$$

$$\ldots.\ FS*\ \ \ \ \ \ .\ 0\ \ldots.,\ 0)) = F_t\ (D_t\ \ D_n\ ,\ FM\ (D_t\ \ldots..\ D_n\ ,\ 0\ 0)\ldots.$$

$$\ldots\ FM\ (Dj.\ .\ D_{\prime\prime}\ .\ 0\ldots..0)) = F'”^{+}”\ (D,\ .\ D_n\ .\ 0\ldots.0).\ \blacksquare$$

Teorema II.9.2. *Sini valabile incluziunile :*

(11.9.7) $Ff>(0\ldots..\ 0,\ 0\ldots 0)\ \ c\ (D_t\ldots\ldots\ldots D_{\prime\prime}\ ,\ 0,\ldots,\ 0)\ c$

$$Q\ F<*>\ (D_t\ \ldots..\ D_n\ .\ D,\ \ \ \ D_{\prime\prime}).\ VA\text{-}\ <=\ .V*.\ j = 17n.$$

Demonstraţie. Să observăm mai întîi că:

$$(D,\ldots.\ D_{\prime\prime}\ \ .\ D_t\ \ldots..\ D_{\prime\prime}) = F_L\ (D_t\ldots..\ D_n\ .\ ,D_1\ldots..\ D_{\prime\prime}) = D,\ .\ Vj = 17n$$

VA e N*. deoarece $(D_r\ldots..\ D_n)$ este soluţia sistemului (11.9.4).

Vom proceda în continuare prin inducţie după *k.*

Pentru *k* = 1 partea dreaptă a relaţiei (II.9.7) revine la :

$$Pi\ (D_t\ldots\ D_n\ .\ 0,\ldots,\ 0)\ o\ F_j\ (D_1\ \ldots..\ D_a\ .\ D_t\ \ D_{\prime\prime})$$

adevărată în baza proprietăţii de monotonie a funcţiilor standard (v, teorema II.9.1).

Presupunem relaţia (II.9.7) adevărată pentru *k.* Avem:

$$F?^{+}«\ (D_t\ldots..\ D_n\ ,\ 0,\ldots,\ 0) =$$

$$= P,\ (£>.\ D_n\ \ \ • P\hat{}\ (D,\ P>_{\prime\prime}\ .\ 0\ldots 0)\ldots.P^{l}n^{kJ}\ (D_r\ \ldots..\ D_n\ ,\ 0,..,\ 0))\ c;$$

$$£\ Pi\ ('>>\ .\ P>_{\prime\prime}\ • P\hat{}\ (\hat{}\ldots\ldots\ldots\ldots\ldots\ D_n\ .\ Di\ \ldots..\ D_n)\ \ \ \ F«>\ (Di\ \ldots\ldots\ldots\ldots\ D_{\prime\prime}\ .$$

$D_1 \ldots D_{\ldots})) = F_f\ (F^{*'}\ (Di \ldots\ldots D_n\ .\ Di \ldots\ldots D_n) \ldots\ldots$ F<*> ($O_1 \ldots\ldots D_n\ .\ D_t \ldots$

•••■ $P^{k>} \ldots\ldots\ldots Dn \bullet D_x \ldots\ldots$ £>„),..., FJ*' $(D_L, \ldots, D_n\ .\ Dl\ D_n)) =$

$$= F^{\wedge}\ (Di \ldots\ldots D_n\ ,\ D_{,,}\ldots,\ D_{\ldots}).$$

Procedăm prin inducţie după *k* şi pentru a demonstra partea din stingă a relaţiei (II.9.7).

Pentru *k — 1* verificarea este banală. Presupunem incluziunea adevărată pentru *k.* Avem :

F*/$^{+1}$' (0…0, 0…0) = F_j (F<'> (0,..., 0, 0..0)….F^1/* (0,..., 0, 0.... .... 0), FJ'> (0,..., 0, 0,..., 0),..., F<*» (0,..., 0, 0,..., 0)) c *F, (*D_t.... D_n Ft« *(D,.. D„ ,* 0…..0)…FJ*^{1} *(*D_t *,..., D„ , 0,...,* 0)) = *F™* (D_x D_n*,0*….0).

Aici am folosit faptul că F*/' (0,..., 0, 0,..., 0) '= *D, ,* V *k* e *N*. *j* = Ln- ■

Teorema II.9.3. *Are loc relaţia* $P_j = D_t$ *, j* = 1, *n.*

Demonstraţie. Ţinînd seama de teoremele precedente şi de monotonia *funcţiilor standard avem :*

$$P) = \bigcup_{\text{fc-i}} P^l P\ (£1\ .\quad -\ 0\text{--}.\ 0) = \bigcup_{\text{fc-i}} (D_t D_n\ ,\ 0\ \ldots\ldots\ldots\ldots 0)\ £$$

£ U $F|^{k}$ *(*$D_1 \ldots\ldots D_{\ldots}.\ D_l \ldots D_n$*)=Fj (D,…* D_n *, D, ... D„) =* D_f*, \/j = Ln.*

Invers, D, = J F™ (0,.... 0, 0,..., 0) £ ţj *F™ (*D_l*. .* D_n , 0,... 0) = t-i

= U F'*'0.....0) = P,. ■
t-i

Teorema 11.9.4. *Limbajul D este de tip* 2 *dacă şi numai dacă el aparţine unei sp-algebre secvenţiale cu număr finit de generatori.*

Demonstraţie. Să presupunem mai întîi că limbajul *D* este de tip 2, generat de gramatica *G = (*I_N *,* I_T *,* x_0 ; F) şi *e £ D* (dacă *e* ^*D* şi *CU* ar fi sp- algebra secvenţială cu D\{e}& *CU.* atunci algebra *CU.'* construită prin adăugarea la sistemul de generatori ai lui *CU.* a limbajelor {e} şi 0 rămîne o sp-al- gebră secvenţială şi *D* s (%').

Conform observaţiei II.7.1 limbajul *D* esle prima componentă a soluţiei minimale a sistemului :

€11.9.8) x, = |J $i_k F_{jk}$ (Xj ..... x„) u IJ (x,.....r„). $j = 1, n.$
$t_i ei_T$ $x n^{e,} N$

Fie deci $D = D_t$, unde (Z), ... D_n) esle soluţia minimală a sistemului de mai sus.

Să considerăm alfabetul ..... disjunct de I şi sistemul de ecuaţii :

$$v, = U\ i^*\ P» \ldots\ldots în) \text{ u u } V.\ G,_k(\& \ldots\ldots <„), j = iTTi,$$
$$fj.e\ r_T \qquad fc\text{-}i$$

care este liniar în Yj,..., $Y„$ pe alfabetul I_T u ..... £„}•

Observînd că limbajele F_{jk} ,..., U ,..., £„) sînt finite urmează
că ele sînt limbaje regulate şi atunci soluţia minimală a sistemului are drept componente limbaje regulate (F_I , P_2,..., $P_„$) (v. teorema II.4.18).

Fie A_h - $(1_T$ u ……. $ţ„\}$, S_D, $f_h;s_{oh})$ semiautomatul iniţial în care lim
bajul regulat este reprezentat prin mulţimea de stări finale W_A, li
$= 1, n. n$ n
în automat ul .4 = j] -/I „ = (/$_T$(J {£, £„}, [] S*. f; s_0 = (s_{01}. s_{02} ..... s_{0B})).

unde (((Sj. s_n), y) = (/\ (s_v y)..$f_„$ (.-.„, y)). Vy E 1_T U {£I 5,}.

(.«,].s_B) e .S'j, limbajele P_h. **/i =** 1, n sînt reprezentate respectiv de
n
mulţimile »=i de stări V\ - {.$■ = (s, ,..., s„), s_t s W_A}.

Să considerăm limbajul $Qs,?* = \{p$ j $p\ e,\ f$ (s, p) **e** V\}.

Eslc clar că $= Qs^v^$ Vom nota cu Q limbajul obţinut din limbajul Q £ (I_T **U {?!** P^{r*n} substituirea simbolilor £t..... $ţ_n$ prin limbajele
D,.... D_a respectiv,.

în baza teoremei II.9.3 avem $D_h = P_h = Q_{S()}$-v_{ll} $h = 1$, n.

■ Vom arăta că sp-algcbra 7/ generată de limbajele Q_s este secvenţială. Avem :

$$= \mathbf{U}\ M\ A<_s.<_Aj.r„\ \mathbf{j} \qquad V'JJU$$
$$e'r$$
$$n$$
$$J\ U\ Qf\ ts.ijJ.\ l'_h\ U^{£}\ îf\ (^S> \quad '*)■$$
$$\mathbf{fc-1}$$

unde

$$_\ ,a\ v\ x\ I\ e.\ dacă\ se\ V\backslash$$

l 0. in caz contrar.

Din relaţiile (II.9.8) şi formulele (II.9.5) urmează că toate cuvintele limbajului $P_„$ încep cu litere din alfabetul I_T şi prin urmare $Q/f^{\wedge}.i^{\wedge}.y,, = 0$ •

Bămîne deci = j i_t U e *(f (s₀. i_t),* V\)].

Apoi :

$$= (\mathbf{J}\ it[Q/^{u\,£}\ (f\ (^{s>}\ '*)\blacksquare\ '\ *)1^{\ u}$$

$$\mathbf{U}\ \mathrm{U}\qquad\qquad \mathbf{U}\ _{s}\ (/\text{-}(s,\ <*).\ V\backslash)].$$

Deoarece $= D_k = P_h = Qs_g.v_k$ avem

$$Q_S.C_A = (J\ h\ \sim Qf(_9.<_k).v_k\ U\ £\ (f\ (s,\ i_t).\ V'*)]\ u$$

U IU i/(ʳj U ᶠ(/(ˢo''A *■))] (Q/(t.i_k).y ₙ U ᵉ *(f (ᵉ> 'ᵗ)'* v *)).

{y G-Jy

în sfîrşit obţinem :

(11.9.9) $Q_t.v_n = Qf(».$ U ᶠ (/ (ˢ> *;)> $Vn)$ U U [Q/r«₀. <zj. r_t U

U eî/˙(s₀, i_z), VJ] [0/f₅.^. v_h **U e (f (s, ij, V_{ft})],**

şi ţinînd seama de relaţiile (II.9.1—3) derivata unui limbaj E din CU aparţine algebrei CU ceea ce arată că sp-algebra CU este secvenţială.

Să presupunem acum că *CU,* este o sp-algebră secvenţială generată de limbajele E\ , E_2E_n unde E_k c *I*, k* — 1, *n.* Vom arăta că oricare limbaj *E* e *CU* este de tip 2. Să observăm mai întîi că *E* = <p (E_z,..., E_n) unde <p este o funcţie standard, 9 : *(<Ţ> (I*))ⁿ -> cp(I*).*

Putem scrie :

$$E_h = \mathrm{U}\ '\ E,\ \mathbf{u}\ \mathrm{s}\ (\mathrm{EJ}),$$

(ei

şi cum *CU* este o sp-algebră secvenţială *d,* E_k e *CU,* deci $c_t\ E_k$ = *t?*$_{ll>}$(E_l ,.. .... E„), unde qs$_{ZA}$ sînt funcţii standard.

Sifftemul de ecuaţii

■T_o = ţ> (Xj ,..., X_n)

•t» = U '<(?<» (*1 >•••> *.) u 8 (D»)J, h = 1, n.

<er

admite soluţia (E, $E_{\pm}$,..., E„).

Mai mult, soluţia sistemului (II.9.10) este unică, ceea ce rezultă din teorema 1.5.9 şi din observaţia 1.5.6 şi deci ea este şi soluţia minimală. De aceea prima ei componentă — limbajul E — este un limbaj de tip 2. ■

Teorema II.9.5. *Un limbaj E* s /* *esle de tip* 2 *dacă şi numai dacă limbajul E esle reprezentat într-un AL* (v. definiţia II.9.2).

Demonstraţie. Fie E un limbaj de tip 2 şi să presupunem mai întîi că *e E.* Să notăm prin *CU,* sp-algebra secvenţială construită în demonstraţia teoremei precedente cu E e *CU-*

Mulţimea finită de limbaje {Q_s, V_s} care generează sp-algebra *CU* este pusă în corespondenţă biunivocă cu alfabetul finit Z prin bijecţia : -*
-> Z, unde

$$^{a}\ (Qs\text{-}\ V_h) = {}^{i}s.V_h.$$

Considerăm automatul de tip Leticevski *AL= (I, cp*$_f$ (Z*), F ; z_s , V_t) unde definim *F* ({z. y },i) = a^{-1} *(3*$_t$ *Qt.v*$_h$*)-* Să observăm că funcţia *F* satisface condiţiile din definiţia II.9.1. Ţinînd seama de expresia (II.9.9) avem:

(II.9.10) F({z,.r_4},i) = {$z_{/rs\text{-}(AFjt}$ u e *(f* (s, i), V\)} u

u UU^{Z}/ rj u e U S (/-(s, V»)J.

Să observăm că limbajul *E* este ie, rezentabil în semiautomatul *A*$_{:}$^ = = *(I, (21, Fq£ ;E)* unde *(E*$_1$*, i) = dfEi, YEi* s *dl. i e I,* prin mulţimea de stări finale {E', j e e E'}.

Bijecţia a : {& , $_F$J s $\overset{R}{e}$ f] S_h , V_h $\overset{n}{s}$ R ^{S}4 "* ^{Z}n-i n-i

poate fi extinsă în mod natural la un homomorfism între sp-algebrele *(dl,..*>U) şi *(cp* (Z*), . ,ţj), care va fi notat de asemenea cu a.

Vom arăta că homomorfismul a induce un homomorfism surjectiv de semiautomate a : ->■ *AL,* deoarece a *(Fc*^ (E_x, i))= *F* (a (E\), i), *YE*$_L$ *dL.*
i e *I,* Pentru aceasta va fi suficient să arătăm că *«.(F*^ ($Q_{s\text{-}Fft}$, /)) *=F (ct.* (Qs.r), O sau <x (d, Q_s,r_A) = E ({$r_{s>Fft}$} , i) ceea ce are loc ţinînd seama de relaţiile (II.9.9) şi (II.9.10). Homomorfismul a : <7^ -» *<p (Z*)* duce un hmbaj care conţine cuvîntul nul *e* într-un limbaj care conţine pe *e.* Prin urmare limbajul E, de tip 2, va fi

reprezentat în automatul *AL.*

Dacă *c* s *E* atunci în automatul *AL* în care se reprezintă limbajul E\{e} adăugăm stării iniţiale cuvîntul nul e şi obţinem un nou automat *AL* în care limbajul E este reprezentat.

Invers, să presupunem că limbajul E = L(AL) unde *AL= (I, cp$_f$* (Z*), *F*; L$_o$). Vom arăta că limbajul E aparţine unei sp-algebre *dl* secvenţiale finit generate, de unde va rezulta, în baza teoremei II.9.1 că limbajul E este de tip 2.

Pentru fiecare *Q <p^r (Z*)* fie limbajul :

$$F_Q = \{p \mid p \text{ e } I^*, e \text{ e } F(Q, p)\}.$$

$$^E t, = \{p|P^{e\ e\ e\ F}(L_o, p)\} = E \text{ şi}$$

Avem $Eg_t \text{ j } _{Q1} = E_{O1} \text{ u } E_{QI}$.

$$^E Q,Q, = F_{Q1}\ E_Q, .$$

$$^E F(Q,() = 3_(E_Q .$$

Ca atare sp-algebra secvenţială generală de limbajele {E{$_z$} | *z e Z}* u U {{e}, *0}* are un număr finit de generatori şi *E* aparţine acestei sp-algebre. ■

§ II. 10. LIMBAJE DE TIP 1 (LIMBAJE SENSIBILE LA CONTEXT)

Vom prezenta în acest paragraf proprietăţile limbajelor de tip 1.

Observaţia II.10.1. Conform definiţiei II.2.1 pentru o deducţie $q_l \overset{a}{\Rightarrow} q_2$ avem / *(qj < l ţq$_2$),* dacă *q$_2$ / e.*

în baza acestei observaţii clasa limbajelor de tip 1 esle inclusă în clasa limbajelor generate de gramaticile descrise în :

Definiţia 11.10.1. O *gramatică expansivă* '> este o gramatică G = *(I$_N$, I, x$_0$; /*')* pentru care orice regulă de generare *(j*| -> r$_2$) G *F* are proprietatea / (r$_2$) *l* (r$_2$) sau, dacă *r$_2$ = e,* atunci *r$_t$* e *I$_N$* şi *r$_t$* nu apare în partea dreaptă a unei reguli de generare din *F.*

Pentru a stabili legătura dintre gramaticile de lip 1 şi gramaticile expansive, este necesară în prealabil :

Teorema II.10.1. *Fie G = (I$_N$, I$_T$* , x$_0$; *F) o gramatică de tip* **1** *care nu conţine reguli de generare de forma x -> e, x* G *I$_N$. Fie* / j , *r$_2$* G (/$_v$ U /$_T$)*, *astfel incit r$_2$ să conţină cel puţin un simbol din I$_N$ şi l (i|)* c *l* (r$_2$). Atunci, *pentru gramatica G$_L$* = (Z$_A$, , *I$_T$* , x$_0$; *F* u {/V -»• r$_2$}) *există o gramatică de tip* 1 *echivalentă.*

Demonstraţie. Dacă *r$_{ſ}$ = x$_2$... x$_m$* şi *r$_2$* = xj *x'$_2$... x'„* cu *x$_}$, x$_k$* G

Z_A . $j = 1$, m. $k = 1$, n, atunci adăugăm alfabetului I_N noi simboli neterminali ■j :m , iar lui F regulile de generare :

Xj x_2... x_m -> Zt x_2... x_m

"1 ^'2 ••• "* *1 '2 "3 ••• •

(R) "1 "2 ••• -m-1 3-m * *1 -2 ••• *m — 1 'MI + i ••• Xw

"1 ~2 ••• zm 'Tm + 1 ••• 'n •T1 *2 ••• "m -rm + t ••• xn

! -^1 %2 ••• 1 zm -rm + l ••• ~* z1 'z2

Deoarece regulile de generare din mulţimea R au aspectul prescris regulilor de generare din gramatici de tip 1, gramatica G' - $(I_N$ u .... "m$\}$. I_T , x_0 ; F u $R)$ este de tip 1. Mai mult, G' este echivalentă cu G_t . Dacă p G L (G') şi în deducţia x_0 => p au fost folosite doar reguli din F atunci evident p G L (G_1).

***> denumirea în literatura engleză este „length increasing grainmar".**

Dacă în deducţia x_0 => p, a fost folosită prima regulă de generare din R o (şi în deducţie ea este obligatoriu prima regulă din R folosită, cuvintele care apar în regulile din F neconţinînd simboli $z)$, atunci *toate* regulile din R apar în deducţie în aceeaşi ordine în care apar în R, deoarece în caz contrar obţinerea unui cuvînt terminal n-ar fi pesibilă.

Astfel, în gramatica Gj avem deducţia x_0 => /j => r_2 => p şi p E L (GJ). ci, G, G,

Invers, incluziunea L(Gj) c $L(G')se$ obţine imediat.

Dacă r_x şi r_2 conţin şi simboli din I_T, adică $r| = y_1$... y_m şi r_2 = $y|$... $y'„$, unde $y_{)t}$ y'_k s I_N j I_T , j = 1, m, k = 1, n, atunci vom adăuga alfabetului I_N noi simboli neterminali { $z_{\#}$ | h = 1,mj}, /Hj m şi {z^ 1*1* = = 1, nj, H] < n, cîte unul de fiecare din simbolii $\{y_j$ j j = 1, m} u $\{y'_k$ / k = l,n} care aparţin lui I_T. Fie F' mulţimea de reguli de generare care se obţine din F u {Fj -» r_2}, în care fiecare y_j şi y'_k terminal este înlocuit prin Zj - ul sau z'_k -ul respectiv.

Gramatica $G[= (1_N$ (J {$z_{>ft}$ | h = 1, mj U $[z_k$/1*1* = 1, nj , I_T, x_0; F' {z_y -» -> y_t / y_j E I_TJ (J $\{z_k$-+ y_k |y_k E $I_T\}$ este o gramatică

echivalentă cu *G, .*

* * *

Intr-adevăr, dacă *p G L(GJ* şi avem .r$_0$ => Tj => r$_2$ => *p,* atunci avem $_e$ • * Oi Gi *Gi* *x$_0$ => . => r'2 => p"* deducţie care se obţine din cea precedentă înlocuind

°*i* ^{G}i °i ----- $_z$ --- simbolii terminali *{y$_j$* I *j* = 1, m} (J *{y$_k$* / *k* = 1, n} cu simboli corespunzători. Această nouă deducţie poate fi completată astfel : .r$_0$ => *p"* => *p,* apli- cînd regulile de generare de forma *z$_j$* -> *y$_t$* sau (şi) *z'$_k$* *y$_k$* ; deci pe *L* (G J.

• •

Este clar că, dacă *p* E *L* (GJ) şi în deducţia *x$_0$* => *p* nu s-a folosit regula de generare *r$_2$,* atunci *p* s *L* (GJ, deci *L* (GJ £ *L* (GJ.

Invers, fie pe *L* (GJ, adică *x$_a$* => p. Fie *i*| şi r$_2$' cuvintele obţinute din ^{c}i' ^{r}i Şi *r2* P™ ^{11} înlocuirea simbolilor terminali cu simboli de forma *Zj* sau *z'$_k$* .

Dacă în deducţia x$_0$=> p nu s-a utilizat regula de generare *i*| -> r$_2$, atunci c, există în gramatica *G$_1$* deducţia *x$_0$=> p,* în care nu apare regula de generare ri -> r$_2$ şi p E *L* (G J.

Dacă T$_0$ => *i*| => r$_2$ =* • *P* • avem • x$_0$ => => *r$_2$* => *p* şi *p* e *L* (GJ. G' C'J GJ 01 C1 C1

Să observăm că gramatica *G*|, echivalentă cu Gj, nu estedetip 1, dar are forma gramaticii Gj din prima parte a demonstraţiei.

Există deci o gramatică de tip 1 G" echivalentă cu G{ , deci echivalentă cu Gj. ■

Observaţia 11.10.2. Proccdînd analog ca în demonstraţia teoremei precedente, se poate arăta că pentru gramatica *G$_2$* = (1^ , *I$_T$* . x$_n$; FţJ {r, -> -> /jr$_m$ -> /',}) cu *l (rj)* s: *l* (rj) şi *rj* conţine măcar un simbol din *j* = 1, *m,* există o gramatică de tip *1,* echivalentă.

Teorema II.10.2. *Pentru orice gramatică expansivă există o gramatică de tip* **1,** *echivalentă.*

Demonstraţie. Putem presupune că gramaticile care vor apare aici nu conţin reguli de generare de forma *x -> e,*

deoarece în caz contrar putem îndepărta aceste reguli şi introduce un simbol iniţial y_0 ; însoţit de regula de generare y_0 -> *e.*

Fie $G = (I_N , I_T , x_0 ; F)$ o gramatică expansivă. Atunci $F = F'\bigcup \{(r> -» -» {}^{r}'i)$ I *l ('fi < l (*r*'j| J — 1»* m}, i^ar regulii de generare din F' satisfac la, restricţiile impuse unei gramatici de tip 1. Dacă $F'' = 0$, atunci şi $L(G) = 0$ deoarece nu există nici o regulă de generare de forma x_0 -> r. în acest caz există evident o gramatică de tip 1, Gj = $(I_N , I_T , x_0 ; 0)$ echivalentă cu *G.*

Dacă $F' / 0$, atunci, ţinînd seama de observaţia II.10.2, există o gramatică de tip 1, echivalentă cu gramatica expansivă *G.* ■

Un alt tip de gramatici legate de gramaticile de tip 1 este dat în definiţiile de mai jos :

Definiţia II. 10.2. O gramatică cu marcaj final # este o gramatică :

$$6^* = (I_N \text{ tj } \{ * \} , Ir , x_0 ; F).$$

astfel încît # $ I_N tj I_T şi fiecare regulă de generare din *F* este de una din formele :

(II.10.1) $r_x \; r_2$, *r! -> *r_2, fj* -> r_2# ,

unde f] , r_2 e (7^(J I_T)* şi 74 conţine cel puţin un simbol din •

Definiţia II.10.3. *O gramatică de tip 1 cu marcaj final #* este o gramatică cu marcaj final #, în care regulile de generare obţinute din (II.10.1) prin îndepărtarea marcajului final #, satisfac restricţiile impuse gramaticilor expansive.

Definiţia II.10.4. Limbajul $L(fi, \#)$ generat de gramatica G* este

$$L(G, \#) = \{p \mid p \text{ s } F_T, \# \; T_0 \; \# \; \#p \; \#\}.$$
$$^{c}\#$$

Remarcăm că $L(G, \#) / L$ (G#) întrucît în $L(fi, \#)$ cuvintele apar fără marcaj final.

Observaţia 11.10.3. Noţiunea de echivalenţă între gramatici cu marcaj final este asemănătoare celei obişnuite. Astfel, gramaticile G^ şi G* sînt echivalente dacă $L(\&, \#) = L(G^2, \#)$.

Observaţia IL10.4. Pentru orice gramatică de tip 1 există, în baza teoremei II.10.2, o gramatică de tip 1 cu marcaj final, echivalentă.

Rezultatul reciproc este conţinut în :

Teorema II.10.3. *Pentru orice gramatică de tip* 1 *cu marcaj final există o gramatică de tip 1 echivalentă.*

Demonstraţie. Fie G# = $(I_N$ U {*}, I_T > ^{r}o < *F)* ^{0} gramatică de tip 1 cu marcaj final #. Putem presupune că nu apar regulile de generare de forma *x -» e* prin îndepărtarea marcajului final.

Pentru fiecare simbol y e I_N (J $I_T = I$ se introduc trei simboli : ♦y, y*. *y+ şi fie :

*/ = {#y | $y <= 7$}, 7# = {y# j y 7}, #7# = {*y* i y s 7}. iar pentru

r = JZ1 <Z_2... $y_{„}$-i ^{n} > 1. ’Ji 7, j = 1, n fie

r = $(*yf)\ y_2 ... y_n$-! y_n, r = $y_x\ y_2 ... y_{n_Y}\ (y*)$,

$_{r}$ = ((*1/1) y_2 - y_n- 1 (y*) pentru $n > 1$
| +y# pentru $n = 1$.

Vom arăta că gramatica expansivă G_1 = (7$_N$|J*7|J 7#(J *7*, I_T , ; FJ,
unde $F_{\pm}$ constă din regulile de generare obţinute prin următoarele procedee :

1) dacă (r_1 -» r_2) s F atunci r| -> r_2, ->■ #r_2, r* -» r* şi *r# _> aparţin lui F_t ;

2) dacă (#rj -> #r_2)^s F atunci -» *r_2 şi #/* -♦ *r* sînt în F_x ;

3) dacă (rjf -»r_2#) eF atunci z* _» /# şi #/* -» #/# sînt în F_x ;

4) # y -* y, y* -» y şi *y* "* P $^{sînt\ în\ F}$*i*, pentru orice y e 7,

unde r_1,r_2 s 7* \ {?}, este echivalentă cu gramatica G# , adică L *(GJ = =• L (G, #).*

Intr-adevăr, dacă *.r* => p există o deducţie

*0 # => *P + > deci L (GJsL (G, #) şi invers, dacă există

o deducţie *.r_0* => p şi L (G, #) c L (GJ.
^{c}i

Conform teoremei II.10.2, există o gramatică G', de tip 1 echivalentă cu gramatica expansivă G_l şi deci echivalentă cu gramatica G*. ■

Teorema II.10.4. *Fie G o gramatică expansivă (de tip* **1).** *Există o gramatică expansivă (de tip* **1) G'** *echivalentă cu G, astfel Incit regulile de generare ale lui G', care conţin simboli terminali, sînt de forma x* **-> i,** *x <s* I_N , *i <=* I_T .

Demonstraţie. Vom înlocui în regulile de generare ale gramaticii G = = *(*I_N , I_T , x_0 ; *F)* fiecare simbol terminal *i* printr-un nou simbol neterminal x_t. Gramatica G' va fi *G' = (*I_N U {^{x}i I ' ^{s} -M> I_T • ^{x}*o F'),* unde *F'* se obţine din *F* prin procedeul indicat mai sus şi prin adăugarea regulilor de generare (x_f -> i); ea este

echivalentă cu G şi de acelaşi tip cu G (exerciţiu !). ■

Definiţia II.10.5. O gramatică de ordinul n, n > 1 este o gramatică G = = *(*I_N| I_T ,x_0*; F),* pentru care V *(*r_Y -> r_2) e *F, l (rj n* şi *l (*r_2*) < n.*

Teorema II.10.5. *Pentru orice gramatică expansivă de ordinul* n > 3 *există o gramatică expansivă echivalentă de ordin n* — 1.

Demonstraţie. Fie $G_n = (I_N , I, x_0; F)$ o gramatică expansivă de ordin *n* > 3, în care putem presupune, fără a restrînge generalitatea, că regulile de generare care conţin simboli terminali sînt de forma *x -> i* (v. teorema II.10.4). Vom construi acum gramatica expansivă G_{n_j} de ordin n — 1 echivalentă cu G_n . Fie deci $G_{a_1} = (I_N , I_T , x_0 ; F')$, unde *F'* se construieşte astfel :

1) dacă $(r_1 \to r_2)$ e *F* şi *l* (r_2) 2, atunci avem şi *l* (rj 2 şi luăm $(r_i\ r2\ )^e$ -F •

2)dacă $(r_\pm \to r_2)$ s *F* şi *l* (r_2) > 3, atunci există $x_4, x_2 , x_3 , x_4 e ,$ astfel ca $r_r = x_L r[, r_2 = x_2 x_3 x_4 r_2$ cu $r\{ , r_2$ e I^*_N (eventual puţind fi cuvinte nule).

în această situaţie distingem două cazuri :

2a) dacă $r_4 = e,$ atunci r_4 = r, şi introducem doi noi simboli neterminali Zj z_2 şi regulile de generare :

$$z_2 , Zj \text{—»} \quad *2\ ^x 3 \qquad F ;$$

2b) dacă $r_1 \pm e,$ atunci $r[= x_5 r_4$ cu $r_4 e F_N$ şi r_x = Xj x_5 $r'f ;$ introducem atunci doi noi simboli neterminali z_3, z_4 şi regulile de generare ;

$$x_4\ x_5 \text{-»}\ z_3\ z_4,\ z_3 \text{-*}\quad \mathbf{-4\ r\ 1\ \sim *\ *r_3}\ x_4$$

3) regulile de generare din *F'* se obţin prin aplicarea procedeelor de la punctele 1) şi 2), după caz, fiecărei reguli de generare din *F.*

4) Ia regulile obţinute mai sus se adaugă neschimbate regulile de generare din *F* de forma x -* i.

Este uşor de verificat că $L (G_a) = LţG\hat{}j).$ ■

Consecinţa II.10.1. Pentru orice gramatică expansivă există o gramatică expansivă echivalentă de ordin 2.

Fie *G* o gramatică expansivă şi *n* = max $\{l (r_2) \mid (r_r \text{-*}\ r_2)$ e F$\}$.

Evident, *G* este o gramatică de ordin n şi prin aplicarea repetată (de *n — 2* ori) a teoremei precedente, obţinem în final o gramatică expansivă echivalentă de ordin 2.

Definiţia II.10.6. O *gramatică liniar-mărginită* este o gramatică $G = = (I_N , I_T , x_0 ; F)$, pentru care, dacă $(r_4 \text{-*}\ r_2)$ *e F,* atunci avem una din următoarele situaţii :

1) $r_4 = x_0 ; r_2 = x_0$ x, x e $I_N ,$

(II.10.2) 2) = Xj x_2 , r_2 = Zj z_2, Xj , x_2, $z,,$ z_2 e 1_N

3) fj = x_4, $r_2 = y$, $y\ I_N$ u I_T , $x_4\ I\#$

$^{x}o\ \{y> .\ ^{z}2 \bullet *\}\bullet$

Teorema II.10.6. *Pentru orice gramatică expansivă G există o gramatică liniar-mărginită G', astfel incit L (G') = L (G) sau L* **(G') =** *L* **(G) \ {e}.**

Demonstraţie. Putem presupune (în baza teoremei II.10.4 şi a consecinţei II.10.1), că pentru gramatica $G = (I_N , I_T , x_0 ; F)$, F conţine numai reguli de generare de următoarele forme :

1) *o -* e>
2) *x-+ y, x e* I_N , *y <=* I_N *u* I_T ,
3) *x* Zj z_2 , unde ij, , :2 e .
4) x, x_2 -> Zj z_2, unde x_1, x_2, z_1, z_2 e 1_N .

Să construim o gramatică liniar-mărginită : G' = *(*I_N u {x^, Xj}, fi > lo < *F'), XQ , x[* fi . în acest scop *F'* va conţine toate regulile de generare ale lui *F* de la punctele 2) şi 4), care satisfac cerinţele impuse regulilor de generare din gramaticile liniar-mărginite. Apoi, pentru fiecare regulă de la punctul 3) vom introduce regula de generare x.Xj -» z_L z_2, iar simbolul xî mai apare în următoarele reguli :

XQ x'o Xi, fi x -» xx{, xx{ -» fix , Vr e (v u I_T . Mai avem şi fi -» x_0 .

Întrucît toate regulile de generare au aspectul prescris regulilor de generare (II.10.2), gramatica *G'* este liniar-mărginită.

Să considerăm acum gramatica G_o , obţinută din gramatica *G* prin eliminarea regulii de generare x_0 -> *e* (dacă această regulă era în G). Avem :

$$L(G_0) = \begin{cases} L\,(\mathrm{G}), \text{ dacă } (x_0 \text{ -» } e) \in F, \text{ adică } e \in L\,(\mathrm{G}) \\ L\,(\mathrm{G}) \setminus \{e\}, \text{ în caz contrar.} \end{cases}$$

Vom arăta că $L\,(G_o) = L\,(fi')$, de unde va rezulta valabilitatea teoremei.

Fie *p* e *L (fi₀),* adică există deducţia x_0 => *p,* în care să presupunem că G. s-au folosit un număr de n reguli de generare de forma indicată la punctul 3), n > 0.

Ţinînd seama de definiţia gramaticii G' din deducţia precedentă, putem

• * obţine o deducţie în G' de forma $X_Q \underset{G'}{\Rightarrow} x'_o x[\underset{G'}{\Rightarrow} X_Q X| \ldots x[\underset{G'}{\Rightarrow} x_0 x\{\ldots xj \underset{G'}{\Rightarrow} p.$ — G*

Cuvîntul x_0 *X[... xî* avînd *n* simboli *x[* serveşte ca un „rezervor" de sim-

boluri xj, permiţînd utilizarea regulilor din G' care înlocuiesc regulile de forma 3) din G_o .

Aici utilizăm, cînd este necesar, comutabilitatea simbolului *x[* cu orice x s fi u fi pentru aducerea simbolilor *x[* pe poziţiile necesare. Avem deci *p e L* (G'), de unde *L* (G_o) £ *L (G').*

Invers, fie p e L (G'), adică există deducţia $x_0 \Rightarrow_{G'} p$.

Obţinem imediat o deducţie în G_o, $x_0 \Rightarrow p$, eliminînd toţi simbolii x_x care apar în deducţia din G'. Astfel L (fi[1]) £ L (G_o). ■

Consecinţa II.10.2. Pentru orice gramatică G de tip 1 există o gramatică liniar-mărginită G', astfel incit :

$L(fi)$, dacă $e \pounds L(G)$ $L(G) \setminus \{e\}$, în caz contrar.

Afirmaţia rezultă imedial din teorema precedentă şi din faptul că orice limbaj de tip 1 poate fi generat de o gramatică expansivă. ■

Observaţia II.10.5. Fie $G = (L_N, L_T, x_0 ; F)$ o gramatică liniar-mărginită şi $p = i_t\ i_2 \ldots\ i_n \ \&\ L(G)$. Există o deducţie $x_0 \Rightarrow p$, care poate fi de- G scrisă cu ajutorul unui hiperarbore, prin analogie cu descrierea prin arbori de deducţie (v. definiţia II.5.1) a cuvintelor generate într-o gramatică de tip 2.

Definiţia II.10.7. *Un hiperarbore de deducţie* pentru o gramatică liniar- mărginită este un hiperarbore care satisface următoarele condiţii :

1) orice nod este etichetat cu un simbol din $I = I_N \cup I_T$
2) rădăcina hiperarborelui este etichetată cu x_0
3) nodurile care au un descendent propriu sînt etichetate cu simboli din J_N .
4) dacă avem în hiperarbore o muchie de forma din figura II.10.1.a, atunci în deducţie se foloseşte regula de generare $x_0 \rightarrow i_0$; dacă în hiperarbore apare o muchie de forma descrisă în figura Il.lO.l.b, atunci, corespunzător, avem în deducţie regula $x_1x_2 \rightarrow z_t\ z_2$, iar pentru o muchie de forma din figura II.lO.l.c se utilizează regula de generare $x \rightarrow y$.

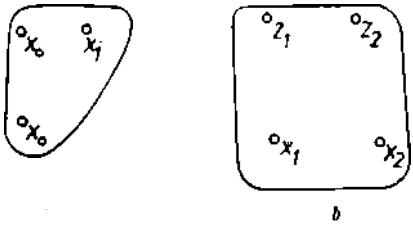


Fig. II. 10.1

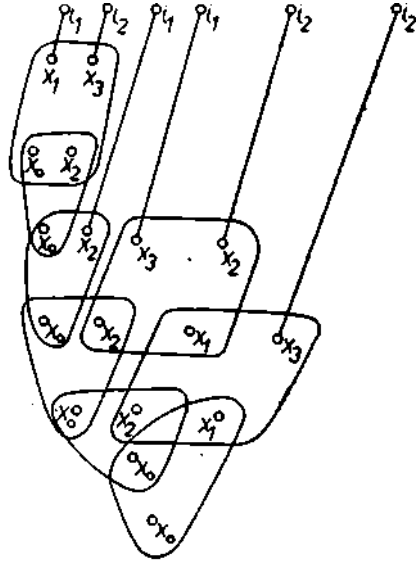


Fig. II. 10.2

5) nodurile terminale ale arborelui în ordinea în care apar descriu cuvîntul generat prin deducţia descrisă de hiperarbore.

Exemplul II,10.1. Fie gramatica G = ({.r$_0$, , .r$_2$, ,r$_3$}, {q , i$_2$}, x$_0$; F),

unde. F = {z$_n$ -» $x_0\ x_x$, $x_0\ x_0\ x_2$, $x_0\ x_2$ -> 24 ,r$_3$, x_0-* -> i_2, x_2 -> i\ ,

x_2 -> i_2, x_2x_l-^x_1x_3, x_2 ij -> $x_3\ x_2$, .r$_3$-> ij, ;r$_3$-»i$_2$}. Hiperarborele din figura II.10.2 corespunde deducţiei .r$_0$ => x$_0$ TJ $X_0\ X_2\ X_L$ => x_0 => $Xg\ x_2\ x_2\ x_t$=> Ty $x_2\ x_3\ x_2\ x_3$ => $Xg\ x_2\ x_2\ x_3\ x_2\ T_3$ => x_t .$r_3\ x_2$ .r$_3$ $x_2\ r_2$ => x_1 .r$_3$ => $x_0\ x_2\ x_t\ x_3$

Remarcăm că lungimea unui cuvînt *p* e *L* (G) este determinată de numărul *m* de aplicări ale regulilor de generare de forma x_0 -» $x_0\ x$ şi fiind anume *m* 4- 1.

Dacă un nod este etichetat cu simbolul x_0 , există uu lanţ care uneşte acest nod cu vîrful .r$_0$ şi în fiecare muchie a lanţului apar două nodyri etichetate cu x_0 . Dacă *u* este nodul aflat la distanţa cea mai mare de vîrful *v,* etichetat cu x_0 , atunci toate nodurile etichetate cu x$_0$ se află în lanţul care uneşte *u* cu *v.*

Ne vom ocupa în continuare de unele proprietăţi de închidere ale clasei limbajelor de tip 1. Amintim (v. teoremele II.2.1, II.2.5 şi II.2.6) că această clasă de limbaje este închisă la reuniune, produs şi închidere prin produs.

Definiţia II.10.8. *O operaţie de substituţie nesingulară generată de mulţimea de limbaje* {Ej ,..., E$_r$} este o operaţie de substituţie (v. def. II.4.3), pentru care *e* E_j, V/ = 1, *r.*

Teorema II.10.7. *Clasa limbajelor de tip 1 este închisă la operaţii de substituţie nesingulare.*

Demonstraţie. Fie $G = (I_N$, , i_2,..., i„}, x_0 ; F) o gramatică de tip 1

şi «Pjtf o operaţie de substituţie nesingulară generată de familia de limbaje *Al* = {E\ ,..., E„}> *Ej* ^{e} > *j* = 1> ^{n}-

Fie $Gj = (I_{N}$, $I_{T)}$, x_{oI}; *Fj)* o gramatică de tip 1, care generează limbajul E_t , J = 1, n, I_{TI} c I_T , *j* = 1, n.

Putem presupune că toate alfabetele neterminale sînt disjuncte două cîte două şi că gramaticile care intervin au regulile de generare care conţin simboli terminali de forma *x* -> i, *x* fiind un simbol neterminal, iar *i* unul terminal (v. teorema II.10.4). Celelalte reguli de generare au forma r' -> *r",* r', *r','* fiind cuvinte alcătuite din simboli neterminali.

Mai putem avea, eventual, şi reguli de forma *x* -» *e.*

Să construim acum o gramatică de tip 1 capabilă să genereze pe *(L* (G)).

* —

în acest scop, fie G' = *(1# , I'_T , x_0* ; F'), unde $I'_x = I_N$ u (J I_{Nl} u {z , /-1 1 e IjJ, $I'_T = \bigcup_{i-i}^{O} I_{Tj}$, >^{ar} *F'* constă din următoarele reguli de generare :

1) x_0 -> *e,* dacă *(Xg-> e) F,*
2) ^1 $^r i$ -» x_2 r_2 şi r_1 -> x_2 r_2 dacă (24 r, -» x_2 r_2) e F,

3) ,r -».r_{OJ} , dacă (x -»i_i) e F,
4) ix -> ix_0 , pentru orice i e I'_T , dacă (x -» i_f) e F,
5) regulile de generare din F_k , F_s F_n .

Observăm că gramatica G' este de tip 1, întrucît ţoale regulile de generare satisfac condiţiile impuse acestui tip de gramatică. Fie p s <p_M (L (G)). Există q e L (G), $q = i_{kl}... i_{km}$ şi q_t ..... q_m cu $q_{„}$ e E_{tft}

astfel încît $p = q_x ... q_m$. Avem deducţiile $rr_0 \overset{a}{\Rightarrow} q$, $x_{oj} \overset{a_f}{\Rightarrow} q$^ .

Să presupunem că deducţia $.T_0 \Rightarrow q$ are aspectul $i_0 \Rightarrow r \Rightarrow$ 7, unde r =- c a G

= $x_{ki}... x_k$ <s r_N , iar regulile de generare care participă în deducţia r =» q m g
sînt de forma x_{tl}-»L , h = 1, m.
• A * A

In gramatica G, avem deducţia $x_0 \Rightarrow$ G x_k $x_{ktffl}...$ x_k .r_{oh} x_k ...x_k

=■ 9l ••• x_{km} => 71 *01 x_k. ■■■ x_{km} 91 ?2 x_t, ••• > x_{km} =* 91 92 x_{0k}.
x_t . ••• x_{km} => G' G'

91 92. Qm = P- deci P e L (G')-

Invers, fie p e L (G'). Există deducţia $x_0 \Rightarrow p$, în care putem folosi re- G'
guli de generare de la punctul 5) şi reguli de generare de la punctele 1)—4). Întrucît simbolul iniţial al gramaticii G' este x_0 în orice deducţie în G' apar în mod necesar reguli de la punctele 1)—4).

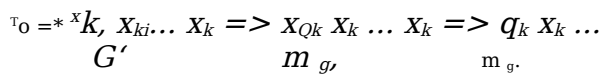
Aplicarea unei reguli de generare de la punctul 5) se poate face numai după aplicarea prealabilă a unei reguli de forma 2) şi 3) sau 4).

Întrucît regulile de generare de la punctele 2)—4) provin exclusiv din regulile de generare din gramatica G. avem corespunzător în gramatica G deducţia x_0 $q = i_{ki}$ $i_{kt}... i_{km}$.

Această deducţie poate fi detaliată astfel :

x_0 x_k, ••• x_{nm} =* G, ••• — 9-

Prin urmare, deducţia $x_0 => p$ poate fi scrisă şi *a-*



Astfel p s <ţ>$_M$ (L (G)), cuvîntul p fiind obţinut prin operaţia de substituţie din cuvîntul q e L *(fi) :* ■

Observajia 11.10.6. Condiţia de nesingularitate impusă operaţiei de substituţie asigură că gramatica G' este de tipul 1. Ea se dovedeşte a fi esenţială, întrucît, aşa CUIJI.'VOBI arăta .ulterior (v. observaţia II.15.1), dacă se renunţă la ea, se poate ieşi din clasa limbajelor de tip 1.

Consecinţa II, 10.3. Clasa limbajelor de tip 1 este închisă Ia endomor- fismele nesingulare ale semigrupului liber I^* (unde prin endomorfism nesingular, înţelegem un endomorfism pentru care *h (i) / e, yi e /).*

Pentru a dovedi aceasta, este suficient să luăm E_{jk} = {? (iy$_{fc}$)}. în acest caz limbajele fiind finite, sînt de tip 3, prin urmare şi de tip 1.

Reamintim necesitatea demonstrării incluziunii $J£_x \ E_2$ (v. observaţia 11.2.2). Avem astfel :

Teorema II.10.lt. *Are loc incluziunea strictă : fi$_x$ J>$_2$.*

Demonstraţie. în baza teoremei II.5.2, pentru orice gramatică G de tip 2 există o gramatică G_x expansivă, echivalentă, deci o gramatică de tip 1 echivalentă. Prin urmare, $J?_x$. Pentru a arăta că incluziunea este strictă, să considerăm gramatica de tip 1, G = ({x$_0$, x_x, y, y_x, y_2, z, z_x, z$_2$, z$_3$}, {*» > 'K > F î ^{T}o)> ^{u}nde F constă din :

$$\mathbf{x_n} \text{ -*} x_o z_x,\ x_0\text{-+ } x_x\ \mathbf{z_x},\ x_x \text{ -> } i_x\ y_x,\ x_x\text{-+ } i_x\ x_x\ y_x\ ,$$
$$yi\ ^z\mathbf{i}\ ^z\mathbf{3 > i/i}\ ^Z\mathbf{3 -> i/}^z\mathbf{3 . </}^z\mathbf{3 -*}\ y^z>$$
$$ih\ y\text{-*}\ yi\ y»\ yi\ 1/2 \text{ -* } 1/1/2,\ 1/1/2 \text{ -* } 1/1/1\ ,$$
$$ZZj \text{ -» } Z_2 x_x,\ z_2\ Zj \text{ -» } z_2\ Z,\ z_2\ z \text{ -> } z_x\ z,$$
$$</\text{-*'}\ ]/>\ z\ G\ .$$

Limbajul generat de această gramatică este :

(II.10.3) L (G) = {i" $i''\ i?$ / n 1}, care nu este un lumbaj de tip 2, dar este un limbaj de tip 1.

Mai rămîne să verificăm egalitatea (II.10.3). Pentru aceasta vom observa că primele patru reguli de generare permit deducerea cuvintelor

$$*_Q \Rightarrow ix\ y''\ ^z i\ .\ m, n > 1.\ a$$

Folosind în continuare celelalte reguli de generare, putem ajunge la un cuvînt din , dacă, şi numai dacă, *m=rt.* Astfel obţinem $x_0 \Rightarrow_G$ ij .

într-adevăr, avem .r$_0$ y" zf => *i% y*$_\pm$ z$_3$ ^î'^{1} =>

G

=* i' $y^{\wedge -1}$ yzz''^{-1} => i; $y''\sim^{2}$ y_x y_2 zz?"^{1} => i'' $y?\sim^{2}$ y_x y_2 z_2 z_x $z^{n}{}_{x}\sim^{2}$ -

=> $y''\sim^{2}$ $yy_2{}^{z}2$ ^{z}i^{z}?$^{-2}$ =* '2 $y''\sim^{2}$ $yy^{*}{}^{z}2^{zz}$?$^{-2}$ => «2 $y''\sim^{2}$ yyi ^{z}2 ^{zz}?~^{2}

-> $i^{n}{}_{x}$ y?-^{2} yyi ^{z}i^{zz}"$^{-2}$ =* % $yi\hat{}\sim^{2}$ $y^{2\,z2\,z}$"$\sim^{2}$ => >2 $y^{n\,z}$" =* '2 £ •

Dacă am avea *m=f= n,* atunci am obţine z_0 => z" *i"* z" *zŢ~tt* , dacă *m > n* sau .r_0 => *i" y"~m i" i" , dacă n > m* eliminarea simbolilor neterminali Zj sau Vi ne mai fiind posibilă.

§ II. 11. AUTOMATE LINIAR MĂRGINITE

Vom studia o clasă de automate care servesc la recunoaşterea limbajelor de tip 1.

Definiţia II.11.1. *Un automat liniar mărginit* este un 5-uplu *ALM* = = (/, S, *f*; s_0 , #), unde *I* este un alfabet finit, # s / este numit *simbolul de marcaj, S* este o mulţime nevidă, finită de stări interne, s_0 e £ este *starea iniţială,* iar *f : S* x *I* x {1,0. -1}), astfel incit, dacă (s', *i, k) «*
e *f (s,* #), atunci i = #, iar dacă *i, k)* e *f (s, i),* atunci *i* = #, este *funcţia de tranziţie* a automatului.

Un *ALM* nu înlocuieşte simbolul # cu vreun alt simbol din *I* şi nu poate înscrie într-o celulă ocupată de un simbol din *I* simbolul #.

Obserua(ia II.11.1. Un *ALM* lucrează în mod nedeterminist. Aflîndu-se în starea s şi primind la intrare un simbol *i* (aflat pe banda de intrare (v. fig. II.ll.l)) automatul trece într-o nouă stare *s',* deplasează eventual banda într-un sens sau altul şi înscrie pe bandă un nou simbol *i'.* Automatul înscrie pe bandă simbolul # dacă, şi numai dacă, primeşte la intrare simbolul #.

Prin urmare, din punctul de vedere al activităţii de înscriere pe bandă a simbolilor din alfabetul /\{#}, introducerea simbolului # pune capăt acestei activităţi.

Dacă («', i', 1) s *f (s, i)* automatul poate deplasa banda de intrare cu o celulă la stingă, după ce a înscris în celula din dreptul capului de citire- scricre simbolul i, dacă (.*•', i', — 1) e *(s, i)* deplasarea se poate executa la dreapta, iar dacă *(s', i',* 0) s *f* (.$, î) banda poate rămîne pe loc.

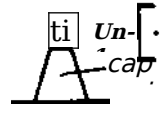


Fig. II. 11.1

Conţinutul benzii de intrare este reprezentat de un cuvînl *p* e /* înscris între două marcaje #.

Activitatea automatului începe cu marcajul sting al benzii de intrare şi cu starea iniţială s_0 . Dacă prinlr-o serie de mişcări ale benzii şi de tranziţii succesive automatul *poale* ajunge în extremitatea dreaptă a benzii şi trece într-o stare dintr-o mulţime dată de stări (numită mulţime de stări finale) cuvîntul aflat pe bandă este acceptat.

Definiţia II.11.2. O *configuraţie instantanee* a unui *ALM = (1. S. f;* s_v , #) esle un triplet 5 = *(p, s, q)* unde s e S, *p, q /*.*

Dacă configuraţia instantanee este *(p,s.q)* automatul se află in starea s, banda conţine cuvîntul *#pq#,* iar capul de citire-scriere se află în dreptul celulei care conţine primul simbol din cuvîntul *q.*

Definiţia II.11.3. *Configuraţia instantanee* 8, = (p_i , , gj *precede imediat configuraţia instantanee* 8_2 = (p_2. s_2, q_2). dacă una din următoarele condiţii este satisfăcută :

1) Vi= 'V Ş*Va ='<{' ‘ar *(*s_2*i',* 0) e *f (*s_x *. i)*

2) Vi= 'V *Pi = Pi*[iar] ('[s]2 > ''• 1) [e] f([s]i ’

3) Pi= *pii* •V!= G V Şi *Pi = P- <ti* = 'i '2 V- iar • G > - >) [e] *f (*[x]*i* . G)-

Vom nota aceasta prin :

in cazul 1) automatul găsindu-se in starea şi citind de pe banda de intrare simbolul *i* poate trece în starea s_2 şi înscrie pe bandă simbolul *i'* în locul lui *i* şi nu deplasează banda.

Jn cazul 2) automatul găsindu-se în starea s_x şi citind de pe banda de intrare simbolul *i* trece în starea ,s_2 şi înscrie pe bandă simbolul *i'* în locul lui *i.* deplasează banda eu o celulă, cuvîntul aflat la stingă capului de citire- scriere, devenind p_1*i'.* în sfîrşil, în cazul 3) automatul aflat în starea Sj primind la intrare simbolul ij trece în starea s_2. înscrie în locul simbolului ij simbloul i_2 şi mişcă banda la dreapta cu o celulă.

Definiţia II.11.4. *Configuraţia instantanee* 8 *precede configuraţia instantanee* 3'. dacă există un şir finit de configuraţii instantanee

$$3 = 8_0 , 8, \quad 8, = 3' \text{ cu } 8,)— 8_{>+1} . j = (\text{Tn} \sim 1.$$

*

Vom nota aceasta 8 |— 8'. Şirul 8 = 8_0 |— 8j |— ... j— $8_{„}$ = 8' îl vom numi *calcul* al configuraţiei 8' pornind de la configuraţia 8, iar *n* va fi numit *lungimea calculului.*

Remarcăm că pentru fiecare configuraţie a unui calcul, lungimea cuvîn- tului conţinut pe bandă rămîne neschimbată.

Definiţia II.1I.5. *Limbajul acceptat de un ALM prin mulţimea de stări finale* $S_($ este :

$$L\ (ALM) = \{p\ p\ I^*.\ (e,\ s_0\ ,\ p)\ H\ (q,\ Sj\ ,\ e),\ s,\ G\ S,\}.$$

Definiţia II. 11.6. *Limbajul pe alfabetul* 1_Y *S I acceptat de un ALM = = (I, S. f;*s_0 *, #) prim mulţimea de stări finale* S_t *esle :*

$$L\ (ALM,\ IJ = \{p \mid p\ e\ I^*\ ,\ (e,\ s_g\ ,\ p)\ \text{ţi}\ (y,\ s,\ ,\ e),\ ,s_t\ f= .\ y\ e\ /^*\}.$$

Evident *L (ALM. I) = L (ALM).*

Teorema II.ll.l. *Pentru orice limbaj E* c I'_T *de tip 1 există un automat liniar mărginit ALM astfel incit E = L (ALM,* I_T*).*

Demonstraţie. Conform teoremei II.10.6, putem presupune că limbajul E\{e} esle generat de o gramatică liniar mărginită *G = (*I_N *,* I_T *,* x_g *; F)* (v. definiţia 11.10.6), adică $(r_t$ -» $r_2)$ e *F* dacă şi numai dacă apare numai una din următoarele situaţii :

1) 'i =*0 ’ ^{r}2 = ^{x}o^x> ^{x} e

2) = l’l *x-2* ’ ^{r}2 ~ ’l ~*2* ’ -^{T}1 ’ *x2* ’*1 ’ “2^S

3) fi =Ti, $r_2 = y$, *y*e I_N (J I_T , x_t e I_N , x_0 $ {y, , $_x$l

Avem *e* e *E* dacă .şinumaidacă, (x_0-» *e)* s *F.*

Să considerăm automatul liniar mărginit

$$ALM = (/_A\ U\ 'T\ U\ \{'\blacksquare=.,\ ^*h\ S\text{-}\ fis_0,\ ^*)\bullet$$

unde 1_N U ^{J}*T* - S= {.s„ , s, . s_2, s_3, s_4} |J {.s^} | (x, *x* - z, z_2) e *F)}.*

Mulţimea de stări finale este {.s_3}, iar funcţia de trecere este dală prin :

(11.11.1) ($\blacksquare$’>o ’ ♦ > 0 ~ *f* (^{S}o ’ *)’

(II.11.2) *(*s_4*,* *, 1) = f(*$*_0$ *,* *), dacă *e* e E,

(II. 11.3) {(•S’o - JJ. 1). (s_0 • *y*> -0} = *f(*^{x}*0* >orice ye/$_A$. IJ

(II.11.4) *(*s_0 *,* Xj, 0) = *f(*s_0 *, y),* dacă (Xj -> *y)*

(II.11.5) *(*s_{It}*,* x_t*,* 1) = *f(*s_0 *,* Zj), dacă (.^ x_2 z_te *F,*

(II.11.6) *(*s_0 *,* x_2,0) = *f(*s_{It} . z_2), dacă (x, x_2 -><s *F,*

(II.11.7) ($s_($, X_Q . — 1) = *f (*s_g *,* .T_o).

(II.11.8) (.S'$_2$, # • 1) — *f* (^{S}J , #) .

(II.11.9) («2 , =, , 1,) = *f* (.<4 ,

(11.11.10 (^{S}3 , '=! . 1) = *f ($2 ,* Zo)-

(11.11.11 (s_0 . x_0 , 0) = *f(*s_3 *,* xj, dacă (x_0 - x_g€= *F,*

(11.11.12 ($^{S}3 >$ # , 1) = *t*

(11.11.13 ($^{5}3$, # > 1) = *f* ($^{S}4 >$ #)

Pentru toate situaţiile care nu apar în lista de mai sus, *f (s, y)* = 0.

Relaţiile (II.ll.l—2) permit verificarea iniţială a faptului dacă *e* s *E.* într-adevăr, să presupunem că *e* e E. Dacă conţinutul iniţial al benzii este # #, în primul pas capul de citire-scriere lasă primul simbol

neschimbat, banda se deplasează la stînga şi automatul trece în starea s_4 . Apoi, conform relaţiei (11.11.13), automatul trece în starea s_3 şi marcajul drept trece în stînga capului de citire-scriere, deci e s *L (ALM, I_T).*

Invers, dacă *e* s *L (ALM)* aceasta înseamnă că există calculul :

(e, s_0 . ##)[—(#, s_4 , *) (# # , s_3, *e),* ceea ce implică *(x_0* -» e) e *F,* adică *e* <s *L* (G) = *E.*

Relaţia (II.11.3) permite automatului să deplaseze banda intr-un sens sau într-altul, menţinîndu-şi starea .s'$_o$ în căutarea unei situaţii potrivite pentru continuarea calculului.

Relaţiile (II.11.4 —6) reproduc respectiv regulile de generare din *F,* înlocuind simbolii din 7_A-ţJ I_T care apar în partea dreaptă a regulilor de generare prin simbolii corespunzători din partea stingă.

Vom clarifica rolul jucat de relaţiile (11.11.7 — 13) prin examinarea modului în care automatul recunoaşte cuvîntul *p* = i_4 i_2 i_4 i_1 i_2 i_2 care aparţine limbajului generat de gramatica liniar-mărginită considerată în exemplul II.10.1.

Configuraţiile succesive sînt reprezentate în figura II.11.2 prin indicarea poziţiei relative a benzii şi capului de citire-scriere.

Se observă deci că relaţiile (II.11.7 —13) sînt legate de regulile de generare de forma .r_0 -».r_0 x_4,.r_0 , .r_4 e I_N în cazul în care i_0 .r_4 seaflă în bandă în contextul x_t... # pentru *h*0. Cuvîntul .r_0 Xj... # este înlocuit prin cuvîntul#1=$^{4+1}$ .r_0 ... #, pentrufiecare utilizare a unei reguli de generare de forma.T_0 -».r_0 x_t, apărînd pe bandă un simbol'=.

Tot aceleaşi relaţii acţionează şi atunci cînd pe bandă se află cuvîntul # = "x„ + pentru n>0, situaţie în care automatul iese prin capătul drept al benzii şi intră în starea finală s_3.

Prin urmare automatul acceptă acele cuvinte din l'_T care sînt

generate în gramatica G, simulînd orice hiperarbore de deducţie din gramatica liniar- mărginită G, pronind de la nodurile terminale ale hiperarborelui spre vîrf. Avem astfel L (G) c L *(ALM,* I_T*).*

Să presupunem acum că p c este un cuvînt acceptai de automatul *ALM,* adică există un calcul în *ALM :*

$$(e, s_0, p) \vdash (q, s_3, e),$$

unde s_3 este starea finală a aulomatului, adică la sfîrşitul calculului capul de citire-scriere trece în dreapta marcajului drept #, iar automatul trece în starea s_3 . ceea ce se poate face folosind numai relaţia (11.11.12). Aceasta înseamnă că în pasul precedent capul de citire-scriere s-a aflat în dreptul marcajului final iar automatul a fost în starea s_3 , situaţia la care s-a
putut ajunge numai prin utilizarea relaţiei (II.11.10).

Continuînd în acelaşi mod, ţinînd seama de relaţiile (II.11.1—13), găsim hiperarborele de deducţie, care ne permite să reconstituim deducţia cuvîn- tului p în gramatica G, deci p e L *(G).* Cazul $p = e$ a fost tratat mai sus, deci L *(ALM,* I_T*)* c L (G). ■

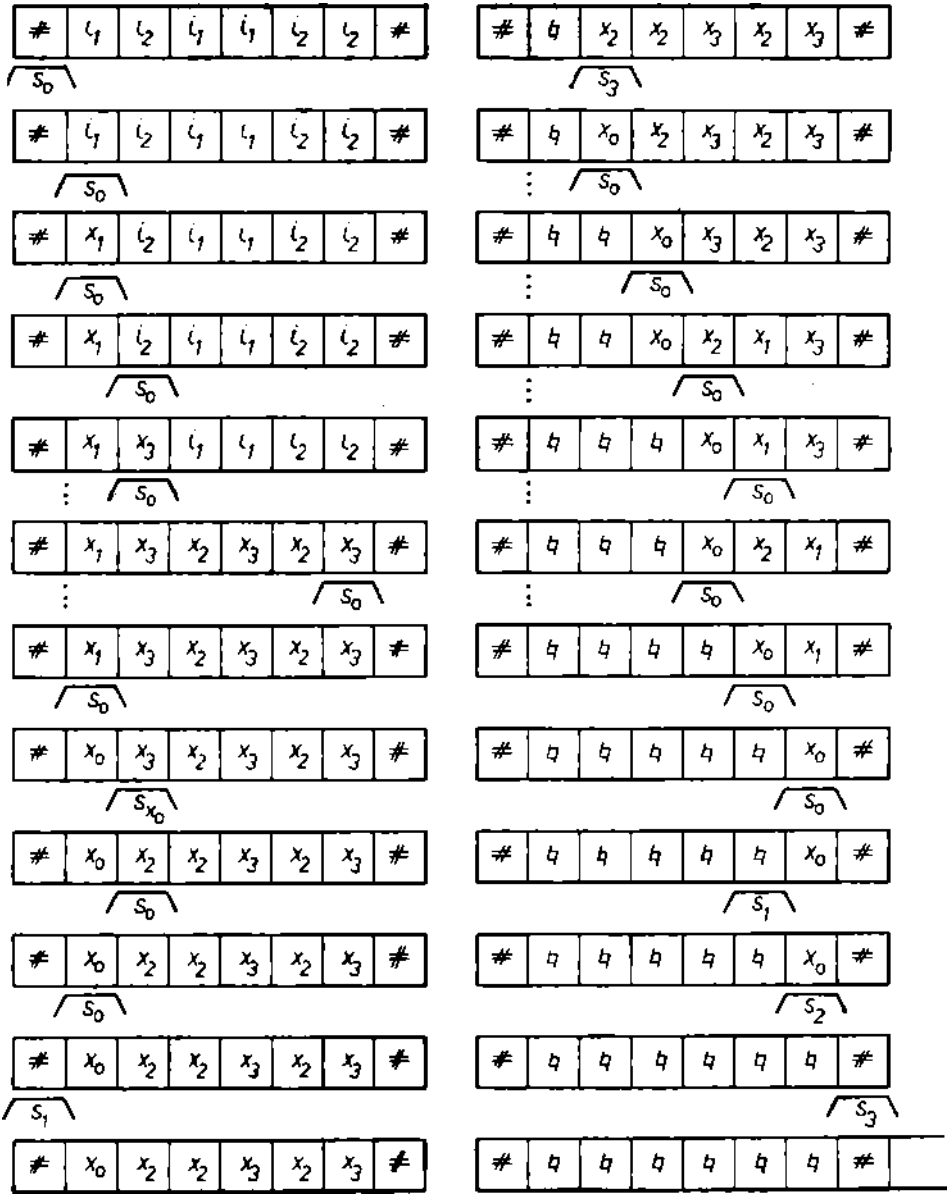


Fig. II.
11.2.

Teorema 11.11.2. *Limbajul L (ALM, acceptat de automatul liniar mărginit ALM = (I, S, f; s_0 , #) prin mulţimea de stări finale S_L c S şi I_1 c I este de tip* **1.**

Demonstraţie. Fie Sj = S, 3.s\ .....s* , A> 1,

(•Sy+i, # > 0) s *f ţsj , #), j* = 1, *k* — 1 şi *(s ,* *> 1) e *f* (■'>'* • #) , -s' e Si},

adică S} este mulţimea stărilor automatului cu proprietatea că automatul, găsindu-se într-o stare din Sj şi având la intrare simbolul #, evoluează prin- tr-un şir de stări fără a modifica conţinutul celulei de pe bandă şi fără a mişca banda, iar în ultimul pas deplasează banda la stînga, trecînd într-o stare finală din S:.

Să considerăm gramatica *G = (I_N, , x_0 ; F)* unde *I_N* este dat de:

1) pentru s e S, *i* s /, *I_N* conţine simbolii .$r_{s(}$, , x# , *x*,
2) pentru *i* ® A *I_N* conţine simbolii *i, i*
3) *1* \ A C *i_N*
4) *I_N* conţine simbolii x_0 şi *x_L* , ,r_0 , x_x *I.*

Mulţimea *F* constă din următoarele reguli de generare :

1) x_4 -»* *i, x_1-»x_1i, Vi G I*
2) *x_0* -»# x#, x_0 .?! x* , dacă (.sj, *y,* 1) e *f* **(5,** i) cu *y* e *I,* sj e S}
3) *x,.$_t$y -* zx$_{ti}$* , *x,,$_2$ y -> *zx,$_f$, *x,.,y** zx* , *x,,$_z$* y* -» *zx* , dacă *(s', y,* -1) G *f(s, i)* şi *z* G *I.*
4) *yx,,$_z$-+ x„z,*yx,,$_z$^*x,$_t$z, yx*-+x,$_t$z*, *yx**x$_{zi}$z*,* dacă *(s', y,* 1) G *f* (s, i) şi z e *I.*
5) - *$x_{s(}$,* x* -> x* , #x# -> *x* , dacă (,s', y, 0) G *f (s, i).*
6) x* -> x# , #x# ->#x*, dacă există *S, k* 1, astfel ca (Si , *y,* 1) S *f (s, i), (s''$_{+1}$*, #, 0) 6 /• *(sf , #), j* = 1, A- - 1, *(s^r*, #, -1) e *f(s';, #).*
7) *x,^-* *x$_{fj}$*, *x* -» *x* , dacă există {s/....?}.'} c S, A > 1, astfel ca (s/ ' -$\underline{y}$' $^{-1}$) g *f (s> 0, (s7+i* > # - 0) s *f* (s7 > *).
j = *k* 1. (s > > 1) e *f* (sit ,)•
8) i* -> *i, Viei*
9) #x,$_4$-> i, *x* -» *i* dacă (s, #, 1) G *f (s_Q , #).* Vom presupune că *f (s_Q ,* #) conţine numai triplete de forma (s', #,1), deoarece oricum ple- cînd de la marcajul #, banda se deplasează la stînga.

10. x_0 -» e, dacă *e e L (ALM, If).*

Gramatica *G* expansivă definită aici generează un limbaj de lip 1, conform Icoremei II.10.2. Fie *E* mulţimea cuvintelor *p* pentru care x_0 => *p* şi *p* nu [G]l-7 conţine simbolii ,r_0 şi .t j, unde cu *G1-7* am indicai folosirea exclusivă a regulilor de generare 1) —7).

Observăm că orice cuvînt *p* E *E* conţine exact un simbol .$T_{J(}$ (cu unul sau doi simboli#). Primul simbol din *p* este marcat la stîngă cu iar ultimul este marcat la dreapta cu iar simbolii intermediari nu sînt marcaţi.

Să considerăm aplicaţia ţ> : *E* (S|J /ţj {#})* dată prin :

$$9\ (P)\ -9\ (f/i\ I/2\ -\ Un) = 9\ (l/i)\ -\ 9\ (</\text{»})\ Ş'$$

si , dacă y_j =

#si , dacă y, = $x_{,(}$

'■? *(Ui)* = i «*, dacă y_f = #

dacă y_j = *T# y_j , în celelalte cazuri.

Cuvinlul <p (p) are aspectul <p (p) = #r_2sr_2 #, *i*| , r_2 s *I*.

Vom ataşa cuvîntului p configuraţia S (p) = (q , s, r_2).

Pentru p_y , p_2 e *E* avem $p_r \underset{G}{\Rightarrow} p_2$, dacă, şi numai dacă

S (p_2) |—5 (pi), deoarece fiecărui pas din deducţia p_x => p_2 îi corespunde un pas în calculul 3 (p_2) |—S (p_t) şi reciproc, făcînd convenţia că şirurile de configuraţii :

('/. 0 H ('/!/> *i - [e]) i— - H (</!/., e) (y, p) şi

(;, s, <7)|— (c, *s'î, yq)*|—...|—*(e, sţ , yq)*|—*(y,s', q)* reprezintă un singur pas în calcul.

Regulile de generare descrise în 1) şi 2) permit deducţiile x_0 => p numai pentru acele cuvinte p cu q> (p) = #*q* şi #, ţe/*, adică pentru acele cuvinte pentru care S (p) = *(q, s,* i).

Aplicarea regulilor descrise în 1) şi 2) presupune că automatul a trecut (deplasînd banda la stînga) într-o stare din mulţimea *S[.*

Folosirea regulilor 8) şi 9) este singura modalitate de a elimina simbolii neterminali din cuvintele deduse.

Să arătăm că *L* (G) = *L (ALM,* 7J. Fie p s *L (G).* Pentru simplitatea expunerii alegem o formă particulară a cuvîntului p, în

cazul general procedeul de demonstrare fiind similar. Fie deci p = i_e i_7 i_8 i_L G 7* şi deducţia :

2) 1) 1) <□) li>

•T_o => x, x* => .?! x_2 x* => Xj i_3 i_2 x* => x_r i_3 $x_{„}$ i_4 i* =>

51) 4i) 4«)

*'s '3 •'«, *l.'f =* *'fl '3 -ᵀs, t. => i, >6 =* ᵀ«. h '■

(e î* => i_s i_7 i_6 i # => 9i) 8) i_8 i_7 i_e i_L.

Am pus în evidenţă regula de generare folosită în fiecare pas al deducţiei indicînd-o pe săgeata dublă.

De exemplu „=>“ înseamnă aplicarea primei reguli de la punctul 4i>).

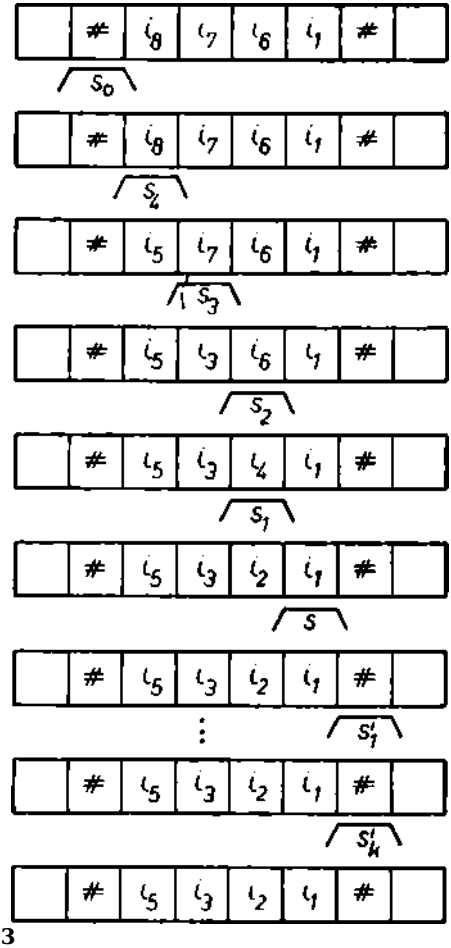


Fig. II. 11.3

/ S' |

Cuvîntului *p* îi corespunde calculul descris în fig. 11.11.3. prin urmare [10]
p s L *(ALM,* Z_r). Dacă *p = e,* există deducţia x_0 => *e* şi conform definiţiei gramaticii *G, p = e e L (ALM,* Z_x). Deci *L (G) £ L (ALM, 1^).*

Invers, fie *p* s *L(ALM,* ZJ. Atunci, ţinînd seama de definiţia gramaticii
G, pentru un calcul al cuvîntului *p* există o deducţie r_0 => *p,* care se obţine *a* parcurgînd în sens invers fazele calculului, deci *p* e *L (G)* şi *L (ALM,* ZJ £ *<= L (G).* ■

Teorema II.11.3. *Clasa este inclusă la intersecţie.*

Demonstraţie. Fie Z.$_t$ şi L_2 două limbaje de tip 1, L, = *L(ALM,* ZJ) şi L_2 = *L (*ALM_2, Z_2), unde ALM_Y = *(1*$_{\pm}$, S_x, ; s_{01}, #), ALM_2 = {Z_2, S_2, f_2; s_{02}, #)> limbajele L_x şi L_2 fiind reprezentate in aceste automate respect tiv prin mulţimile de stări SJ şi S_2.

Vom presupune S\ Q S_2 = *0* şi fie *I* = Zj Q Z_2 .
Să considerăm automatul liniar mărginit *ALM* = (ZtJ (Z_t x Z_2), S_xtJ S_2tJ U {«o >^{s}*i'S*$_2$*'S*$_a$*}, f;s*$_0$,(#,*)), unde {s_0 , , *s*$_2$, s_3} Q (S_t |J *S*$_2$) = 0, iar
f este dat în tabelul II.11.1.

Tabelul II.ll.l

Nr. crt.	(s> ')	*f (s, i)*
1	*))	(«, , (*, *), 1)
2	(s, , *i),* Vi 6 *I*	(«i, *(i. '),* 1)
3	(s, , (#, *))	*), - 1)
4	(s_2 . (i, 0), Vi E *I*	(s_2 , (i, i), -1)
5	(s_2. (*, #))	*), 0)
(>	(s; (ij, i_2)), cu (s', i,) = *I*\ (s, , Â)	(■$^{s'}$l > 0[> *'2^'* *2 ^{s} ^2
4	#)), cu *G', #)* =	
8	- Ats;, *, t),e s; (■$^{s'}$j » 01 * *2)), V 0) ' *2) ^{e} Jj X	*(S_3, (#, *).* -1) («3 ' ('1. <2), -1)
9	*(»3> (*> #))*	(S_{02} , (#, *), 0)
10	(s", (ij , i_2)) cu *(s",* i_2) = *f 2 (^{s} > '2 > k)*	(s", (ij , i_2), *k), i* j *=

Să presupunem că în automatul *ALM* avem iniţial pe bandă un şir de simboli din Z. Conform posibilităţii descrise la punctul 2 al

tabelului, automatul înlocuieşte simbolii *i s I* cu perechi de forma *(i,* z), deplasînd banda spre stînga.

Acest proces se termină cînd se întîlneşte marcajul (#,#) din partea dreaptă. La întîlnirea acestui marcaj automatul trece în starea s_2 şi deplasează banda la dreapta pînă la întîlnirea marcajului (#.#) din partea stingă, automatul fiind pus pe starea s_{01}.

în acest moment automatul începe să opereze asupra primilor simboli ai perechilor scrise pe bandă.

Cuvîntul pj alcătuit din aceşti simboli este din limbajul Lj = *L (ALM,* Zj) dacă, şi numai dacă, automatul ajunge în starea *s'* s Sj , în dreptul marcajului (#, #) din partea dreaptă. Deci, dacă *p* s L_t automatul trece în starea s_3 şi menţinîndu-şi această stare deplasează banda la dreapta, pînă cînd ajunge la marcajul (#, #)din stînga. Starea $s_3$este înlocuită acum cu starea ,s_{02} şi se fac calculele care simulează cele executate de automatul ALAL, avînd acelaşi cuvînt de intrare *p.*

Automatul *ALM* ajunge într-o stare din S_2 c S_2 dacă, şi numai dacă *p* s L_2. în definiliv automatul *ALM* ajunge in mulţimea de stări finale S_2 dacă şi numai dacă *p e* L_1 Q L_2. Conchidem că limbajul reprezentat în automatul *ALM* prin mulţimea de stări S_2 este limbajul Q *L,* şi deci acesta este de tip **1.** ■

§ II. 12. MAŞINI TURING — DISPOZITIVE DE RECUNOAŞTERE A LIMBAJELOR

în acest paragraf vom studia dispozitivele capabile să accepte clasa limbajelor de tip 0.

O maşină Turing esle alcătuită dintr-un *dispozitiv de comandă* care se poate afla într-o anume stare dintr-o mulţime de stări S, prevăzut cu un *cap de citire-scriere* care preia şi transmite informaţiile de pe o bandă (fig. 11.12.1).

Această bandă este împărţită în celule, are o primă celulă în stînga şi este infinită la dreapta. Celulele pot conţine cîte un simbol dintr-un alfabet finit Z = {ij,..., i„} Q {b}. Vom presupune că în orice moment numai un număr finit de celule, începînd cu prima celulă, pot conţine simboli din Z\{b}, restul celulelor conţinînd simbolul b.

La un moment dat, capul ce citire-scriere citind simbolul din celula din dreptul căreia se află, starea automatului se schimbă, se

înscrie un simbol pe bandă în aceeaşi celulă, după care banda rămîne pe loc sau este deplasată la dreapta sau la stînga cu o celulă.

Descrierea de mai sus este formalizată în :

Definiţia II.12.1. O *maşină Turing* este un 5-uplu : *MT = (I, S,* unde *I* este un alfabet finit, S o mulţime finită, nevidă, numită *mulţimea*

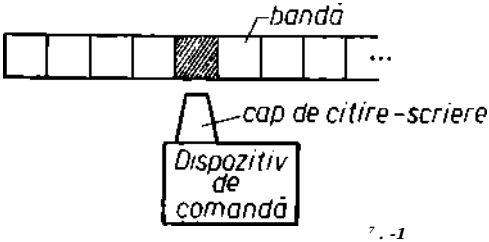


Fig. II.

stărilor, f: (S x /)' *S* x {7 \ {b}} x { — 1,0,1} este *funcţia de trecere, (S* x 7)' c S x 7 este mulţimea perechilor de forma (.s-, i) pentru care funcţia *f* este definită, b 7 este *simbolul „blanc",* iar s_0 este *starea iniţială* a maşinii. Dacă (s, i) *(S x I)'* perechea (s, *i)* este o *situaţie de oprire* pentru *M T.*

f'■ (S x *1)'^ S* x {7\{b}} x {-1, 1}.

Observaţia II.12.1. O *MT(MTN)* funcţionează în mod determinist. Dacă *f (s,* i) = *i',* 1), atunci *MT* găsindu-se în starea *s* şi citind simbolul *i,* trece în starea *s',* înlocuieşte pe *i* cu simbolul *i'* şi deplasează banda la *stînga* cu o celulă; dacă *f (s,* i) = ($', i', 0), banda rămîne *pe loc,* iar dacă *f (s, i) = (s',i',* —1), banda se deplasează la *dreapta* cu o celulă, automatul acţionînd în rest la fel ca mai **sus.** Pentru *MTN* sînt posibile numai deplasări ale benzii la dreapta şi la stînga, banda neputînd rămîne pe loc.

Definiţia II. 12.2. *O configuraţie instantanee* a unei *MT* este un triplet y = ($_s$, *p, n),* unde s s S este starea curentă a maşinii, *p* £ (7\ {b})* este conţinutul porţiunii iniţiale din bandă, pînă la primul simbol blanc, iar n este numărul de ordine al celulei în dreptul căreia se află capul de citire.

O configuraţie instantanee = (Sj, p_x, *nf) precede imediat configuraţia instantanee* y_2 = (s_2, p_2, n_2) dacă, presupunînd că p, = i_L i_2... $i_{„}$, atunci avem una din situaţiile :

1) *f(*^{s}*i* , '„) = (*$_2$. O O. 1 < n,

^2 ■ ^1 4“ 1' *Pi* ~ ⁷1 "■ ⁷»i-l ⁱ⁷n>+1 "• *'n* >

2) *f* («i, i_n.) = *(*s_2*, i,* -1), 2 «: Uj *n,* $n_2 = n_1$ - 1, $p_2 = i_r$... *i^-y i'n.+i •••'»;*

3) *f (*s_1*,* b) = *(*s_2*, i.* 1), *rij = n* + 1,$n_2 = n_Y$ + 1, $p_2 = p_x$ *i ;*

4) *f* (Sj, b) = (s_2, *i,* -1), Jij = n +1, n_2= n_x - 1, $p_2 = p_2 i$ *;*

5) *f* (^{s}>'»,) = (^{s}2 > '• °)> 1 < nj < *n,* $n_2 = n_L$, $p_2 = i_L$... $i_{„i_1}$

[1] '„i+i •••>

6) *f (s,* b) = (s_2, *i,* 0), Hj = n + 1, $n_2 = n_t$, $p_2 = p_Y i$ *;*

Vom nota acest lucru prin :

Configuraţia instantanee y' *precede configuraţia instantanee y",* dacă există un şir finit de configuraţii numit *calcul*: y' = yi |— $y_2$1— ••• I— y„ = y"> ceea ce se va nota cu :

r— Y"

Definiţia II. 12.3. *Limbajul acceptai de* o .1/7' = (/. S. ***f*; b** x_0) *prin mulţimea de stări finale* S_t c S este :

L (-V *T) = {p* I *p e [I*\ {)>}}*, (.ţ$_0$. *p.* 1) p (x, *q,* zi,). x e . *q <= /*}.*

Observaţia 11.12.2. Limbajul acceptat de *MT* este alcătui! din acele cuvinte, care plasate pe bandă, începind cu prima celulă (automatul găsindu-se iniţial în starea x_0). conduc la o stare din , activitatea maşinii începînd cu prima celulă.

Observaţia II.12.3. în continuare vom presupune că, dacă s s Sj *f (s, i) =* (..., 0) adică, dacă maşina intră într-una din stările finale, banda se opreşte.

Exemplul II.12.1. Fie .1/ 7\ = ({0, 1, *x, y, b}.* (x_0 , Xj , x,. x_3, s_4 . x_5}, *f*; b, x_0) unde *f* este dat în tabelul II.12.1.

Tabelul II.12.1

Nr. crt.	s	*i*	/•(s.O
1.1	s_o	0	(Si , 1)
II.1	s_1	0	(s,, 0,1)
11.2	s_1	*y*	(’i , *y,* i)
II.3	s_1	1	(s_s, *y,* -i)
III.l	.s\,	*y*	(3,
III.2	s_2	*X*	(S_3,x, 1)
III.3	s_2	0	(S_4 , o, -1)
IV.l	s_4	0	(s_4,0, -1)
IV.2	s_4	*X*	O_o,x, 1)
V.l	■s_3	*y*	(s_3, *y*> i)
V.2	■'3	b	(s_5, *y.* O

iar (S x *I)'* = {(s_0,0), (x, , 0). (Sj , *y), (s*$_1$, 1), (x_2 , *y),* (x_2, *x),* (x_2,0), (x_4.0) (s_4 , *x),* (x_3, *y),* (x_3 , b)}.

Să presupunem că pe bandă este înscris cuvîntul 000111.

Configuraţiile succesive prin care trece maşina sînt date în

tabelul II. 12.2.

Tabelul II.

1	Tact	1	2	3	4	5	6
2	Configurație	$(s_0, \bar{0}00111,1)$	$(s_1, x\bar{0}0111,2)$	$(s_1, x0\bar{0}111,3)$	$(s_1, x00\bar{1}11,4)$	$(s_2, x0\bar{0}y11,3)$	$(s_4, x\bar{0}0y11,2)$
3	Regula folosită		I.1	II.1	II.1	II.3	III.3

1	7	8	9	10	11	12	13
2	$(s_3, \bar{x}00y11,1)$	$(s_0, x00y11,2)$	$(s_1, xx\bar{0}y11,3)$	$(s_1, xx0\bar{y}11,4)$	$(s_1, xx1\bar{y}11,5)$	$(s_3, xx0\bar{y}y1,4)$	$(s_2, xx\bar{0}yy1,3)$
3	IV.1	IV.2	I.1	II.1	II.2	II.3	III.1
1	14	15	16	17	18	19	20
2	$(s_4, x\bar{x}0yy1,2)$	$(s_0, xx\bar{0}yy1,3$	$(s_1, xxx\bar{y}y1,4)$	$(s_1, xxxy\bar{y}1,5)$	$(s_1, xxxyy\bar{1},6)$	$(s_3, xxxy\bar{y}y,5)$	$(s_2, xxx\bar{y}yy,4)$
3	III.3	IV.2	I.1	II.2	II.2	II.3	III.1

1	21	22	23	24	25	26
2	$(s_2, \bar{x}xxyyy,3)$	$(s_3, xxx\bar{y}yy,4)$	$(s_3, xxxy\bar{y}y,5)$	$(s_3, xxxyy\bar{y},6)$	$(s_3, xxxyyy\bar{b},7)$	$(s_5, xxxyyyy\bar{b},8)$
3	I[illegible]	[illegible]	[illegible]	[illegible]	V.1	V.2

In tabelul 11.12.2 s-a indicat prin supraliniere simbolul din celula din dreptul capului cilire-scrierc. Examinînd acest tablou, concludem că cuvîntul 000111 face parte din limbajul acceptat de maşina Turing dată, prin mulţimea de stări finale $S_L = \{s_5\}$.

Exemplul II.12.2. Să considerăm *M* T_2 = ({0, 1, b}, {s_0 , s_1} x {0, 1, b}, *f;* b. (s_0 , b)) avind mulţimea de stări finale {(s, , b)}. Funcţia *f* este dală de tabelul II.12.3:

Tabelul II.12.3

Xr. cri.	**s**	*i*	**/■(s. 0**
1.1	**(«0 ■ '»)**	**0**	**(M), 0,1)**
1.2	**(«0 - >•)**	**1**	**((Sj, 1), 1, 1)**
II.1	**($S_£$, 0)**	**1**	**((ˢi>0), 1,1)**
11.2	**(«i, 1)**	**0**	**((S, . 1), 0,1)**
111.1	**(Si, 0)**	**b**	**(($_{s1}$,b), 0,-1)**
III.2	**(s,, 1)**	**b**	**((Sj, b), 0, -1)**

Mulţimea stărilor acestei maşini este compusă din perechi de forma (s, *i).* Rolul componentei a doua din o stare este memorarea simbolului de intrare citit de pe o bandă atunci cînd maşina se află în starea (s_0 , **b).**

Trecerea în starea finală (Sj, b) se produce atunci cînd maşina aflîndu-se într-una din stările (s_t, 0), ($s_£$, 1) citeşte semnul b de pe bandă. în aceaslă situaţie se produce memorarea simbolului b, înlocuirea lui cu 0 respectiv 1 şi deplasarea benzii la stingă.

Limbajul recunoscut de această maşină este *L* = {01" | *n* G N} ţj {10"' | m e N}.

Definiţia II. 12.4. *O maşină Turing bilateral infinită* **este** un 5-uplu *M TBI* = = (Z, S, *f*; b, s_0), în care *I, S, f, b, s_0* au semnificaţiile din definiţia II.12.1 iar banda maşinii este nemărginită în ambele sensuri.

Pe bandă este înscris un cuvînt *p* G **(/ \ {b})*,** iar în celulele situate Ia dreapta şi la stînga cuvîntului se află blancuri.

Definiţia II.12.5. *O configuraţie instantanee* a unei *MTBI* este un triplet y = (s, *p, n),* unde s G S este starea curentă a maşinii , *p <= (I* **\ {b})*** este cuvîntul înscris pe bandă, iar *n* este numărul de ordine al celulei în dreptul căreia se află capul de citire-scriere, numerotarea începînd cu celula în care se găseşte primei simbol din cuvîntul *p.*

Fie Ţi > I2 două configuraţii instantanee. Sensul expresiilor y_t I

— Ya Şi

Yi |—y_2 este acelaşi ca în definiţia II.12.2 dacă $n > 1$.

Dacă *f (s, i)* = (s_x, *i*, , - 1), atunci *(s, ip,* 1) (s_x, i_2 *p,* 0), dacă *f (s,* **b) =** = (.s^ , ij , 1), atunci *(s, p,* 0) |— (Sj, ij *p,* 2) şi, în fine, dacă *f* (s, b) = (.Sj , , — 1), atunci (s, *p,* 0) |—(.s'j, *p,* 0), cu ij / b.

Definiţia II.12.G. *Limbajul acceptat de o MTBI =* **(Z, S,** *f;* **b. .?$_0$)** *prin mulţimea de stări finale* S_t £ S este :

I. (M TBI) = {p |p <s (I| **{b})*, (s_0 ,** *p. m)* **H (■'>■■** *<1- s <= S, . </ I*}.*

Maşinile Turing bilateral infinite au aceeaşi capacitate de a accepta limbaje ca şi maşinile Turing, ceea ce se precizează în :

Teorema II.12.1. *Un limbaj este acceptat de o MTBI dacă, şi numai dacă, este acceptat de* o *M T.*

Demonstraţie. Deoarece o *MT* poale fi privită ca o *MTBI* la care capul de citire-scriere nu parcurge niciodată partea infinită din stingă cuvîntului scris pe bandă, este evident că un limbaj acceptat de o *MT* este acceptat de o *MTBI.*

Fie acum *E* un limbaj acceptat de maşina Turing bilateral infinită *MTBI* - (/, S, *f;* **b, s_0)** prin mulţimea stărilor finale *S'.* Construim o maşină Turing *M T = (I_t,* **; (b, b),** s_{01}) care va accepta acelaşi limbaj. Alfa betul de intrare I_t = *I* x (/tj {•}) ^{es}^^{e} alcătuit din perechi de simboluri. Ne putem imagina că banda de intrare a maşinii *MT* are două piste care conţin cîte una din componentele fiecărei perechi *($i_±$, i_2) ~ I_x* ■ Pe prima pistă se află înscris conţinutul benzii maşinii *MTBI,* începînd cu celula din dreptul capului de citire-scriere şi continuînd la dreapta, iar a doua pistă va conţine simbolul ! în prima celulă urmat de conţinutul benzii maşinii *MTBI* situat la stînga capului de citire-scriere, scris în ordine inversă. De exemplu, dacă banda maşinii *MTBI* are conţinutul din figura II.12.la, atunci banda maşinii *MT* are aspectul din figura II.12.1.1). Vom lua = = {(^{s}> t)' (^{s}, 4-) I $^{s\ s}$ S} U {^{s}oi} Ş' $^{vom\ nota}$ l°^{c} de (z\ , b) pur şi simplu i_1 . Funcţia de tranziţie *f* este dată de tabelul II.12.4.

Dacă în momentul iniţial *MTBI* deplasează banda la dreapta înscriind în locul simbolului de intrare *i* simbolul i' şi trecînd în starea *s', MT* înscrie pe banda sa cu două piste (<', I) şi trece în starea *(s',* |), ceea ce indică faptul că *MT* va urmări simbolii situaţi pe pista superioară a benzii.

Cazurile III.l şi III.2 arată că maşina Turing *MT* operează pe pista superioară aceleiaşi modificări ca acelea efectuate de *MTBI* în partea dreaptă a simbolului iniţial.

l0	4	^2	' 3	2	<5	...												
r-/	' -1	' -2	<3	^4	<5		-5	<■- 4	L-J	' -2	<1	r\ ' 0		' 2	’S	' 4	' 5	

Tabelul II.12.4

Xr. cri.	Evoluţia maşinii *MTBI*	Evoluţia maşinii „W*T*
I	f (*a > ') = (' >;i)	A = ((s', t), (E, !), 1)
II	f (*o • ') = (' E, -o	fi (So, - ((«', D, (E, !), D
III.1	f(s. i) = (s', ; ii	f, ((*. î), G i', E')) = ţ(s' , î), (E, i"), 1),' " 1
111.2	f(x, i) (s', 1 -11	fi ((•', î). (i, = ((V. î), (E, i"), -1), E'=fc l
IV.1	/(*, '") = E, 1)	A(GS 1), (G E') = ţ(s', 1), (i.o, -n
IV.2	(« f (s, E') = E, -1)	A((s, 1), (G E')) = ((V, 1), (GE), 1)
V.1	(s /■(•% 0 = ", 1) (»',	/•> ((■', î), (G !)) = fi ((«, 1), (G !)) = - ((*', t), (i " I). 1)
V.2	f(s', /•(s, 0 = E', -1.)	fi ((«, t)> (G 1)) = fi ((•', 1). (G !)) - = ((«', 1), (E', !), 1)

Situaţia este similară pentru deplasările la stînga ale benzii maşinii *M TBI,* în acest caz maşina *MT* prelucrînd simbolii de pe pista inferioară şi depla- sîndu-se în sens invers (v. cazurile II şi IV.1.,2).

Dacă la un moment dat în dreptul capului de citire-scriere al maşinii *MTBI* se află iar celula iniţială, atunci maşina *MT* prelucrează simbolul de pe pista superioară sau inferioară, după sensul în care se mişcă banda maşinii *MTBI,* banda lui *MT* mişcîndu-se totdeauna la dreapta.

Mulţimea de stări finale ale lui *MT* este: S_1 = {(s, ţ), (s, ţ.) | s s

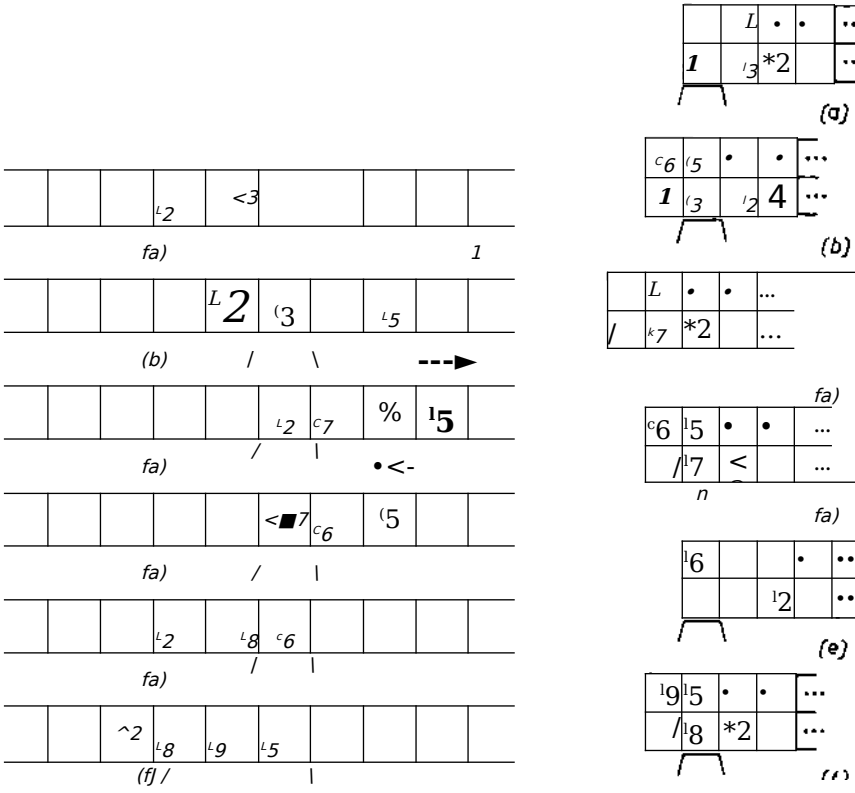
Să arătăm că limbajul acceptat de *MTBI* prin mulţimea de stări *S[* coincide cu limbajul acceptat de maşina *MT* prin mulţimea de stări *S'.*

Fie *p* e *L(M TBI).* Avem calculul în *MTBI: (s₀,p,n)*|—(s, *q,* n_4) cu s s Sj, *q* s *I*.*

Pentru simplitatea expunerii vom presupune că *p = V vom* detalia calculul astfel : (s_0 , i_4 i_2 i_3 i_4 i_5,4) |— (s_4 i_1 i_2 i_a i_e i_3, 3) |— (s_2, ij i_2 i_7 i_e i_5 2) I— *(% , ii* i_2 '7 'a '$_5$. ^{3}) l~ (^{s}4 > ii *'2 h* , ^{4}) |— U ii *h i» '»*5* > 5) cu s s S_4. Diversele situaţii prin care trece *MTBI* sînt descrise în figura II.12.2.

Pentru *MT,* corespunzător situaţiei din fig. II.12.2.a, avem situaţia iniţială reprezentată în fig. II.12.3.a. Configuraţiile prin

care trece succesiv maşina Turingsînt : (s_{01}, *(i_4,* !) (z_5, *i_3)* (b, *i_2)* (b, ij, 1) |— |), (i_6 , !) (i_5, i_3)
(b, i_2) (b, ij, 2) H ((s_2,1), (i_6, !) (f_5 , i_7) (b, i_2) (b, ij, 3) |— (*$_3$ 4). G’e > 0 ('5 . i?) (^{b}> '2) (^{b}> 'i)> 2) I— ((s_4 , !), (i_6 , !) (i_5, i_8) (b, i_2) (b, i\), 1) |— ((s, f), (i.»I) (i_8. i.) (K i_2) (b, ii), 2).



I-ig. II. **Fig. II.**

Corespunzător, configuraţiile succesive parcurse de maşina Turing unilaterală au fost reprezentate în figura II.12.3. Avem astfel p G L(MT), deci *L (M TBI) <= L(M T).*

Incluziunea inversă se poate arăta în acelaşi mod. ■

O altă variantă a maşinilor Turing se obţine prin înlocuirea benzii unice şi capului de citire-scriere unic cu o mulţime de benzi prevăzute fiecare cu cîte un cap de citire-scriere.

Difiniţia 11.12.7. *O maşină Turing cu n benzi* este un 6-uplu *MTMMB - = (n, I, S, f*; **b,** s_0), unde *n* esle numărul de benzi, infinite în ambele sensuri, *I* este un alfabet finit, S o mulţime finită, nevidă, numită mulţimea stărilor, f; (S x I x I x ... x I)'-♦ S x I' x ... I' x {-1, 0, l}^{n} este func- » ' "x ţia de trecere (cu *I'* = / ţj {!>}), *(S* x *1* x ... x /)' £ S x / x ... x *I,* **b** s *I* este simbolul „blanc**, iar s_0 este starea iniţială a maşinii.

Pentru o *MTMMB* dispozitivul de comandă este prevăzut cu n capete de citire-scriere, cîte unul pentru fiecare bandă şi

diferitele benzi se pot mişca în diferite sensuri. Vom conveni că iniţial prima bandă conţine sim- bolii cuvîntului ce trebuie analizat, celelalte benzi conţinînd doar simbolul blanc.

O configuraţie instantanee a unui *MTMMB* este un 2n + 1 - uplu v = = *(s, p_r,..., p_n ,* $m_T m_n$), unde s s S este starea curentă a maşinii, p_t e G (/ \ {b})* este conţinutul benzii *j,* iar n_t este poziţia capului de la banda *j* socotită de la începutul cuvîntului p_j , *j* = 1, *n.*

Definiţia 11.12.8. *Limbajul acceptat de o MTMMB* = (ii, **Z, S,** f; **b,** s_0) *prin mul(imea de stări finale* Sj G £ este :

L (MTMMB) = {p [p e *{I* | **{b}}*,** (s_0 , *p, e,..., e, k,* 1.... 1) p (5, <⁄„ , >•••, k_n), *s* e Sj}.

Teorema 11.12.2. *Un limbaj esle acceptat de o MTMMB dacă şi numai dacă el este acceptai de o M T.*

Demonstraţie. Deoarece orice *MT* poate fi concepută ca o *MTMMB* cu n = 1, este limpede că limbajele acceptate de o *M T* sînt acceptate şi de o *MTMMB.*

Vom arăta acum că un limbaj acceptat de o *MTMMB'estc* acceptat de o *M T.* Să presupunem pentru fixarea ideilor că *n* = 3.

Vom construi o *MT* cu o bandă bilateral infinită avînd 6 piste (în general cu 2/i piste) cîte două pentru fiecare bandă a maşinii Turing cu mai multe benzi, *MT* = (Z(J {1}, *S', f';* b, s_0). unde 1 *I, S' = SI*³ şi* convenim să luăm *se = s,* Vs G S.

Vom folosi în continuare notaţiile :

$$(\beta i) = \begin{cases} i, \text{ dacă } [3 = \\ e, \text{ 1 dacă } 0 \end{cases}$$

$$p_{„} = \begin{cases} I^{1} + Y\text{- dacă } 0 = 1 \\ y, \text{ dacă } 0 = b. \end{cases}$$

$$\begin{cases} f\ Y, \text{ dacă } 0 = 1. \\ \text{şi } 0_Y\{ \\ (0, \text{ dacă } p = b. \end{cases}$$

Dacă *f* (s, , z_2 ? 1*3) = ($» h > ^2 > *3 ’ > ª2 ’ ª3/)» e 0» *J* = vom defini *f' (s* (0j *if)* (0_2 i_2) (0_3 i_3), (Yi > 4 > Ya > 4 - Ys »’a)) în tabelul II.12.5. Aici am considerat, la punctul 2, unul din cazurile posibile (04 = — 1, a_2 = = a_3 = 1), celelalte tratîndu-se analog. Anume, am considerat situaţia indicată în figura II. 12.-1 pentru *MTMMB.* Evoluţia corespunzătoare pentru *MT* este descrisă în

tabelul II.12.5.

Tabelul II.12>5

1

Nr. crt.	Situaţie	Valoarea lui *f* (s (P, i_1)(P_2 i_2) (0_3 i_a), ('(> 'i > Y> <2 > Y> Q)
1	3l 71 ■ p2 T' 72 73 < 3	(s ((Pa + Yi> >1) «Pa + Ya) 'a) ((Pa + Ya) '3>>
2	Pi + Yi + Pa + Ya + Pa + Ya = ■'	(Yi«'i>Ya.'2> Ya-'â)-[1])
	a) max {(aj / *j* = 1,3} = = 1 şi, de exemplu	(■' (Pi + Yi - "i Yi) 'i) ((Pa v Ya ~ "a Ya) 'a (Pa + Ya - [a]3 Ys) '3).
3	a_2 y_2 = «$_3$ Ya = [1] Pi = T Yi = Ya = Ya = [0]	(7i > *1 » , b, i_3), 1) (s (PJ I\), (0, >' , 1, i'_2, 1, ij), -1)
4	Yi=#I	(s (P_x G), (0, i;, Ya ,'2. Y3 --D
5	Yi = I	(s, (0, ij, y_2 , i'_2, y_3 , i'), - 1)
6	Yi = 0	(s, (l,ij,y_2,i',y_3,i'), 0)

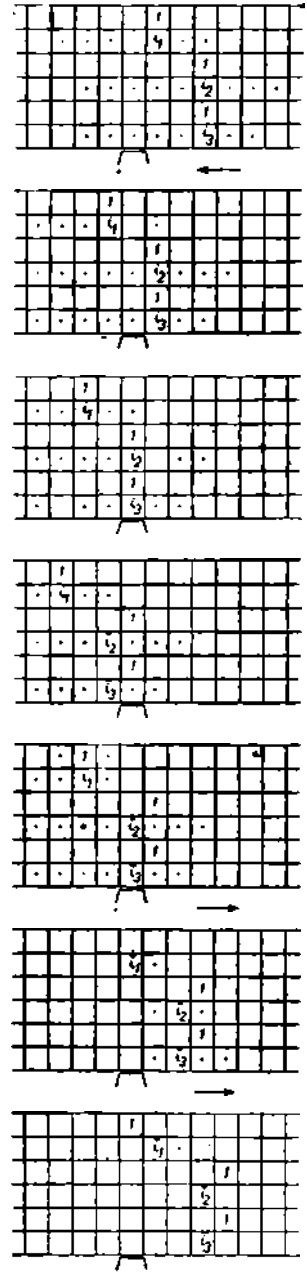
Să observăm (v. fig. 11.12.5) că la un pas al *MTMMB* corespund mai mulţi paşi executaţi de *MT* şi anume: banda maşinii Turing *MT* se mişcă astfel incit prin dreptul capului de citire-scriere să treacă toată porţiunea în care apar simbolii 1 pe pistele de număr impar, starea schimbîndu-se succesiv: *s, si_1 , si_T i_2 i_3* pentru a memora simbolii deasupra cărora se află 1. Capetele de citire-scriere ale maşinii Turing cu mai multe benzi se află tocmai în dreptul acestor simboli ; în pasul următor aceşti simboli sînt înlo- cuiţi cu i_r, i_2 şi i_3 respectiv, iar capetele de citire-scriere îşi schimbă poziţia. Noua situaţie a maşinii *MTMMB* se va regăsi descrisă pe banda cu şase piste a maşinii *MT*. într-adevăr, în continuare maşina *MT* schimbă ultimii simboli i_2 şi i_3 în i_2, i_3 , mută banda la stînga, înscrie pe pista a 3-a şi a 5-a simbolul 1 (pentru a marca noile poziţii ale capetelor 2 şi 3) şi apoi deplasează banda la dreapta.

Apoi, ajungînd în dreptul lui i_x îl schimbă în i_L, starea maşinii a redevenit s şi simbolul 1 este şters. în sfîrşit *MT* mută banda la dreapta, înscrie pe prima pistă 1 şi trece în starea s, noua stare a maşinii *MTMMB*. Astfel, dacă un cuvînt aflat pe prima bandă aduce *MTMMB* într-o stare Sj e Sj, acelaşi cuvînt înscrie pe prima pistă a benzii *MT* (celelalte piste conţinînd numai simbolul b) aduce maşina *MT* tot în starea .

Deci *L (M TMMB)* c *L(MT).* Incluziunea inversă se arată analog. ■

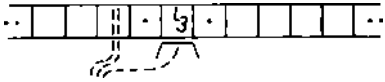
Definiţia II.12.9. O *maşină Turing nedeterminislă (MTND)* esle un 5-n pl ii

$$MTND = (7, S, f;b,s_0)$$



•|-H-|-| | | | IT-

1..... , , , ,



Dispozitiv ul de

în care 7, S, I», s_0 au aceleaşi semnificaţii ca în definiţia II.12.1, iar *f*: (S x *I)' -+<Ţ>*(Sx (İ* **\ {b})** x { — 1,0,1}) este funcţia de trecere.

Fig. II.

Fig. II. 12.5

O *MTND* are mai multe posibilităţi de alegere pentru noua stare în care licee, pentru simbolul care înlocuieşte simbolul prelucrat şi pentru deplasarea benzii. Noţiunea de configuraţie instantanee coincide ce cea introdusă pentru *MT.*

Definiţia 11.12.10. *Limbajul acceptai de o MTND* = (7, S, *f*; b, s_0) *prin mulţimea de stări finale* S, s S este :

L {MTND) = {p | p e (Z\ {b})*, 3(.s_0 , *p, k)* H (.s-, . $_7$, *n).* s 6;}.

Teorema II.12.3. *Un limbaj este acceptat de o MTND dacă şi numai dacă e acceptat de o M T.*

Demonstraţie. Deoarece orice *M T* poate fi concepută ca o *MTND,* esle clar că un limbaj acceptat *de o MT* este acceptat şi de o *MTND.*

Vom arăta acum că un limbaj acceptat de o *MTND* este acceptat de o *MT.*

Fie *p* (/X^fb})* un cuvînt înscris pe banda *MTND.* La prelucrarea fiecărui simbol din *p* maşina *MTND* are la dispoziţie un număr finit de noi evoluţii (schimbarea stării, înscrierea pe bandă şi deplasarea benzii). Fie *r* = max {| *f (s, i)* | s e S, *i* G 7}. La prelucrarea întregului cuvînt există, în total, cel mult succesiuni posibile de configuraţii.

Să considerăm o maşină Turing cu 3 benzi :

$$MTMMB = (3, 7ţJ\ S \times I \times \{-1, 0, 1\}, S, f_Y; b, .s_0).$$

Pe prima bandă a maşinii *MTMMB* se înscriu simboli din *1.*

Banda a doua posedă trei piste destinate respectiv înscrierii simbolilor din mulţimile 7, S şi { — 1, 0, 1], iar cea de-a treia bandă va conţine simboli din alfabetul *I.*

La depunerea unui cuvînt *p* e (7 \ {b})* pe prima bandă *MTMMB* „măsoară" lungimea cuvîntului *p* şi apoi înscrie pe banda a doua toate evoluţiile succesive posibile ale maşinii *MTND* care corespund cuvintelor de lungime *l (p),* două şiruri de evoluţii fiind separate prin blancuri.

în continuare, pentru fiecare şir de evoluţii, se copie cuvîntul *p* pe banda a treia şi, sub acţiunea simbolilor înscrişi pe banda a doua, *MTMMB* „simulează" pe banda a treia acţiunea *MTND,* înregistrînd şi schimbările de stare comandate de tripletele înscrise pe banda a doua.

Cuvîntul *p* este acceptat de *MTND* dacă şi numai dacă există un şir de evoluţii care conduce la o stare Sj e Sj, deci, dacă şi numai dacă *MTMMB* intră la un moment dat în starea . Astfel, dacă *p* esle acceptat de *MTND,* în mod necesar maşina cu mai multe benzi ajunge la un nou şir de evoluţii înscris pe banda a doua care o conduce la o stare e Sj şi reciproc. ■

Teorema II.12.4. *Un limbaj acceptat de o MT este de tip* 0 *; L (MT)* G *J>$_u$. Demonstraţie.* Fie *MT* = (7, S, *f;* **b,** .s_0) o maşină

Turing şi *L (M T)* limbajul acceptat de ea prin mulţimea de stări finale Sj G S. Putem presupune fără a restringe generalitatea eă {S_x x **{/\ {b}}}** Q (S x /)' = 0, unde *(S x 1)'* este domeniul de definiţie al funcţiei *f.* (v. şi observaţia II.12.3). Vom construi o gramatică de tip 0, care va genera limbajul *L(MT).*

Fie *G = (I_N* **. /\ {b},** x_0 ; F), unde = (7 x *I)* |J S IJ {.r_0 , x_x , x_2}, *I =* = (Z\ **{b})U {«},** iar *F* este dat de :

1) Xp —» X_x ,

2) x_x -> *(i, i)* Xj, Vi s *1* | **{b} ;**

3) -Ti ->;

4) x_2 ->(e, **b)** x_2;

o) a $_2$ —> e ,

6) ,s (i, *ij) -> (i, i_2)* .s' pentru orice *i* G *I, s* G S, i_x G *I,* astfel îneît *f (s,* ij = *(s', i_2,* 1) ;

7) _ (i!, i_2) s (i_3, i_4) -* s' *(ij , i_2) (i_3, i_s)* pentru orice i_4 , i_5, *i_2* G e I,ij,i_3 e Ī, s e S, astfel îneît *f* (.?, i_4) = (.■>', i_5 . — 1);

8) (ij . *i_2) s* -» sij s, .s- (i\ , *i_2) -> s i_x .s şi s -> e* pentru Vi$_x$ *I, i_2 I* şi s G Sj .

Să arătăm că L(AZÎ') £ *L* (G). Fie *p = i_2... i_n* G *L(MT).* Să presupunem că în procesul recunoaşterii cuvîntului *p, MT* foloseşte un număr de *m* celule situate la dreapta celulei ocupată iniţial de simbolul i„ . Atunci,

•

utilizînd regulile de tipul 1) şi 2) din gramatica *G,* obţinem x_0 => s_0 ('i > 'i) (i_2, i_2)... (i_n , i_{fl}) Xj, unde *i_j* G **[\ {b},** *j* = 1, *n.* Apoi, folosind regula 3), regula 4) de *m* ori şi, în fine, regula 5), obţinem :

x_0 =^> s_0 (<; , ij (i_2 *, i_2)... (<n , i„) (e,* **b)'".**

în continuare putem folosi doar regulile de forma 6) şi *7),* care simulează activitatea maşinii Turing pînă ce se ajunge la generarea unui cuvînt care conţine simbolul unei stări finale, din mulţimea S_x. Să observăm că acest proces nu afectează primele componente ale perechilor *(i_f, ij).* Vom arăta, prin inducţie, după numărul *t* de. tactc în care acţionează maşina *MT* că, dacă (,s_0 , q *i_2... i_n* , 1)(s,ij,... *i'$_t$, li),* atunci :

(II. 12.1) SQ *(ij, i^ (i_2 , i_2) ,..(i_n , i_n)* **(e, b)^m =^>** *(ij , ij) (i_2, i_2) ... (i_k j, i'$_k$ j) Sj (*»> '*) ••• (in + m > ^n + m)*

cu ij,...,i_ns I| {b}, i_{n+1} = ... *i_{B+m} = e,* ij,..., i'$_{n+m}$ G *I* şi ij$_{+l}$ = *...i'„$_{+m}$ =* **b.**

Afirmaţia se verifică în mod evident pentru *t* = 0. S-o

presupunem ade-

* vărată pentru / — 1

tacLe şi fie (s_0 , *ij... i_n* , 1) |— (.s-$_L$, *i[... i'_k , j_ℓ) |— (s_2,* ij'••• i», *j_2)* ^{0} evoluţie a maşinii Turing executată în Z lacle.

Prin ipoteza inductivă avem *s_0 (ij,* i_x)... *(i_n , $i_{„}$) (e,* **b)“** =s> *(ij,* i_x)... - $^{A'}$i (G, . G>) ••• (G + m ’ G+m)- ^{Punînd} .72 = Ji + * (unde as {-1,1}) avem *f (s_l . i'_{jt})* = (,s_2,$i_{\#}$", a), de unde folosind corespunzător 6) sau 7), după cum a = 1 sau a = —1, obţinem s_1 (G,, *i'_{jl})-> (i,, , i'_{J}* sau

(G,-i • G.-i) ’i ('A ’ G.) $^{-s2}$ (G.-i > G.-i) (G, $^{’’h)}$

G = G' ’ y.7’.7 71 • Astfel s_0 (i_r, G) ... ($i_{„}$. $i_{„}$) (e, b)’^{B} =^> (i_T, i}') ...

... s_2 *(i_{Jt}, ... (i_{n+m} , i'_{n+m}),* verificîndu-se afirmaţia iniţială.

Dacă Sj e S_1 aplicînd reguli de generare de tipul 8) în (II.12.1) obţinem : ^{s}o (G, G) (G. G) ■■■ *(G ,* G) («. {})"’=> (G. G) (G > Gb- (G-i, G-i) *1 (G > G) ••• (G+m; G+m) (G ’ G) (G; G) •••• (G-i , G-i)^{5}i G ^{s}i (G+i >

G + l) ••• (G+m ’ G+m) (G > G) (G • G) ••• (G—2 ’ G-î)^{s}i G-i ^{A}i ^{A}’i G -G G G+I

G (G+2 *’ i't+t) - (’n+m > ’n+m) =>* G - G • Astfel *p = i_l i_2... i_n* e *L (G).*

Invers, fie *p* e *L (G).* Vom arăta că *p e L(MT).*

Să observăm că pentru ca *p* s *L* (G) in mod necesar deducţia *x_0 => p* se

• • * *

împarte in trei : a_0 => *r => q => p,* unde în deducţia ,r_0 => *r* se aplică exclusiv regulile de generare de tipurile 1) —5), în r => *q* se folosesc numai reguli de

*

generare de tipurile 6) —7), iar în deducţia *q => p* utilizăm numai regulile de generare de tipul 8). Cum regulile 6) —7) nu schimbă prima componentă a oricărei perechi (G , ij), rezultă că *r* are în mod necesar forma :

r = s_0 (iy, G) • •• (G > G) (^{e}> b)^{m} unde m poate fi ales arbitrar, convenabil pentru partea a doua a deducţiei. Cuvîntul *q* va avea forma *q* = (G, i{)... ••• ^{A} (G ’ G) ••• (G + m * G+m) CU i_{B+1} ... G + m ” C. G G+m ^{S} *I* -*0 G+l ••• ••• = G+m = b> *$^{ar\ A}$ *= Sj. Să arătăm, prin inducţie, după numărul *t* de aplicări ale regulilor de generare de lip C>) sau 7) că dacă există deducţia :

■G (G - G) - (G > G) (e, >•)"’ (G - G') - (G-i, G'-i) * (G, G') - (G+m • '»+»>)•

atunci avem in *AI T* calculul (.s_0 , i_1 ... i_a , 1),— (.s, i{' ... i*, A). Pentru / = 0 afirmaţia este banal adevărată.

Să presupunem deci afirmaţia adevărată pentru / — 1 şi fie deducţia s_0 (r\ . G)- (G ’ G) (^{e}A)^{m} => (G , i_1)... s (G-i ’ G-i) (G > G) ••• (G+m > G+m) =*■ (G > G) ••• (G-i ’ G-i)^{s} (G > G) •••• ••• (G+m > G+m)> în $^{care\ s_ad}$ aplicat *t* reguli de generare, ultima regulă de A generare aplicată fiind regula de tipul 6) : *s" (it-y,* G-i) (G-i • G-i) ^{A}- In * ^z baza ipotezei inductive există calculul (,s_0 , z\ ... i_B , 1) |— *(s",* *i'î* ... G-2G-1 > A- - 1) şi deoarece *f (s",* G-i) = (s, i'_t', 1) găsim (s_0 , ... i_B , l)H(s,G' - ... G'-i. A).

Similar se procedează dacă ultima regulă de generare aplicată ar fi de tipul 7). Prin urmare, deducţiei *r => q* îi corespunde în maşina Turing cal-

cuiul (s_0 , *r*, 1) |— (s, *q, h).* Cum aplicarea regulilor 8) nu se poate face decît dacă s ^ Sj , rezultă că p e *L(MT').* ■

§ II. 13. MAŞINI TURING —DISPOZITIVE DE CALCUL

Un alt rol jucat de maşinile Turing este acela de a „calcula[11] valorile unor funcţii finite pe semigrupuri libere cu valori în semigrupuri libere.

Fie *MT* = (7, S, *f*; **b,** s_0) *o* maşină Turing. Funcţionareaei, descrisă de funcţia *f,* se poate reprezenta şi prin indicarea unei mulţimifinite *Q*deşi ruri care conţin cinci simboluri.

Astfel, dacă *(s,i) (S x I)'* şi *f (s,* i) = ($', i', a), avem *sis' i'* e *Q.*

Dacă două şiruri *sis' i' ct* şi Sj $i_l s'_1$ ij aus = şi *i = i*$_1$,atunci*s'*=sj, *i'* = , a = ai, deoarece maşina Turing funcţionează în mod determinist şi perechea (s, *i)* determină în mod unic tripletul *(s', i', a.) = f (s, i).* Deoarece mulţimea *Q* şi funcţia *f* se determină reciproc în mod unic putem da maşina Turing şi sub forma *MT = (I, S, Q;* **b,** .s_0). Mulţimea *Q* va mai fi numită *programul maşinii Turing.*

Vom nota configuraţiile instantanee (s, p, n) şi prin *qsr,* unde *l (q) = = n* — 1 şi *p = qr,* punînd astfel în evidenţă poziţia capului de citire-scri- ere, el găsindu-se în dreptul celulei ocupate de primul simbol din cuvîntul r.

O configuraţie instantanee *qsr = qsii^* (sau *qs)* este *configuraţie instantanee terminală,* dacă nu există vreun 5-uplu din *Q,* avînd pe primele două locuri *si* (respectiv sb). Starea s este în acest caz o *stare terminală.*

în cele ce urmează, pînă la afirmaţia contrară, vom presupune că toate maşinile Turing considerate au o singură *stare terminală.*

Definiţia II.13.1. *Func(ia calculată pe I* de maşina MT=(I, S, Q* **; b,** s_0) este aplicaţia *F: (/*)' -» I* dată prin

F (p) = dacăs_0 p |— *qsr, s^Sşiqsr* este o configuraţie instantanee terminală (nedefinită, dacă nu există un astfel de calcul,

unde (/*)' £ *I.*

Vom introduce o nouă funcţie prin eliminarea simbolului b, care nu este un purtător propriu-zis de informaţie.

Definiţia II. 13.2. *Funcţia calculată* pe [(7 \ **{b})*]**^{B} *de maşina MT = = (I, S,* **Q;b,** s_0) este aplicaţia F^a*:* ([(7\ **{b})*]")' -»** (7 \ **{b})*** dată prin

$F^n(p_1, ..., p_n)$ = *q,* dacă *F (p{* bp$_2$ b ... bp„) = *uqi*> cu *u, i*> **{b}*,** nedefinită, în caz contrar, unde, ([(7 \ {b})*]")' £ [(7 \ {b})*]".

Definiţia II.13.3. O *funcţie parţial calculabilă de o maşină Turing* este o aplicaţie *h* **: ([(/ \ {b})*]")' -»** *(1* **\ {b})*** pentru care există o maşină Turing a cărei funcţie calculată F^n coincide cu *h,* pe domeniul de definiţie al lui *h.*

Aplicaţia h este *total calculabilă,* dacă este total definită.

Rezultatul central al acestui paragraf este faptul că orice funcţie parţial recursivă pe un alfabet *I* este calculabilă de o maşină Turing. în acest scop este necesară în prealabil considerarea unor maşini Turing simple care calculează anumite funcţii parţial recursive speciale pe un alfabet *I* şi introducerea operaţiei de compunere a maşinilor Turing.

bie 3b T_v = *ţl,* ,...J *$n—2* >* $}, Q_x > b, s_{ox}) şi *Al* Z $_2$ = (Z, {sgg , s_x $s_{m_}$2, s}
Q_2; b, s_{02}) două maşini Turing cu acelaşi alfabet de intrare *I* şi aceeaşi stare terminală s.

Definiţia II.13.4. *Maşina Turing* obţinută prin *compunerea maşinilor Turing* MT_1 şi MT_2 este maşina $MT = Al\ 1|^* Al\ T_2$ = (Z, S, *Q* ; b, s_{01}), unde S = S_X|J S_2 cu S_2 = *{s_{n+1}* iar programul *Q* constă din:

i) şirurile lui Q_x în care starea terminală s este înlocuită cu starea s_{n+1};

ii) şirurile din Q_2 în care s_{02} este înlocuit prin s_{n+1}, iar *s'}* prin s_{n+x+1} starea terminală s rămînînd neschimbată.

Teorema II.13.1. *Fie p, , qj e* Z*, *j* = 1, 3 şi $p_1 s_{01}$ q_t p. $p_2 s q_2$ $p_2 s_{02}$ q_2 MT, MT, p_3 *sq*$_3$. *Atunci* p_t s_{01} q_L /— p_3 *sq*$_3$, *unde Al T* = MT_l* MT_2.

Demonstraţie. în baza definiţiei II.13.4, în maşina *MT* avem calculul

Pi^{5}oi?il—*P2*^{s}*n+i*(*l2*- Apoi schimbînd şirurile din Q, care participă la cal- -vr cuiul p_2 s_{02} q_2 /— $p_3 sq_3$, conform aceleiaşi definiţii, obţinem $p_2 s_{n+1} q_2$ /— $p_3 sq_3$, MT, . MT, de unde p_x s_{01} $p_3 sq_3$.

Prezentăm în continuare unele maşini Turing simple care calculează anumite funcţii parţial recursive speciale pe un alfabet Z. Pentru simplitatea expunerii vom lua *I* = {b, i}, cazul general tratîndu-se analog.

Exemplul II.13.1. Maşina Turing care „transportă'1 blancul este maşina *T = (I,* {s_0 , s_x, s_2 , s_3 , s_i ,s_5,*s}, Q* ; b, s_0), unde *Q* =

{.s_0 bs$_x$ b 1, s_x bs$_2$ i 0, *$s_2$is$_2$il, s_2* bs$_3$ b — 1, *s_3 is$_i$* b 0, s_4 bs$_5$ b — l,$s_5$is$_5$i — 1, *s_5* **bsb** 0}.

Avem *s_0* bbi" b |— sbi" bb. într-adevăr, avem succesiv s_0 bbi^B b bs$_x$ bi" b |—

bii" s_2 **b |— bi" s_4 bb** $\overset{T}{}$ **s_5 bi" bb |— sbi" bb.**

Exemplul 11.13.2. Maşina Turing care deplasează la dreapta conţinutul benzii este maşina *D*~ = (Z, {s_0 , s_x, s}, *Q* ; b, s_0) unde *Q* = {s_0 bs$_x$ b — 1, s_x is$_x$ *i* — 1, s_x bsbO}. Această maşină efectuează transformarea bi" s_ob |— sbi" b,

deoarece avem bi" s_0 b bi" ^{1} s_x iḃ |— bs$_x$ i" b |— s_x bi" b |— $\overset{D}{}$ sbi® b.

Exemplul 11.13.3. Maşina Turing care deplasează la stînga conţinutul benzii este maşina Z)$^+$ = (Z, {s_0 , s_x, s}, *Q* ; b, s_0), unde Q={$s_o$bs$_1$bl,

SJ *is$_l$ i* 1, .s-j b.s'bO). Avem s_0 bi" b I— bi" sb. deoarece $s_{(l}$ bi" b |— bs-j *i^n* b |— bi" Sj b B +

|— bi" sb.

Exemplul 11.13.4. Maşina Turing care permută două cuvinte de pe bandă este maşina *P* = (/, {s_0 , s, ..., s_{18} , s}, *Q* ; b, s_0), în care *Q* va fi descris în

continuare. Această maşină realizează calculul bi" s_0 bi'" b |— bi™ sbi" b. $\overset{*}{}$

Vom nota cu *Q* (Af *T)* şirurile din mulţimea *Q* a maşinii *XIT.* Fie *m* > 0. Vom include în *Q (P)* şirurile din *Q (D$^+$).* în care vom schimba pe s cu s_2 .

Obţinem astfel calculul bi" s_0 bi™ b |—bi" bi™ s,b.

Mai departe, considerăm şirurile *s_2* bs$_3$ b — 1. s_3 *is$_4$* bt). s_4 bs$_5$ b — 1. s_5 is$_5$ i — 1. s_5 bs$_8$ iO, in urina aplicării cărora putem continua calculul, obţi- nînd : bi"bi™s_2 b |— bi"s_5 bi™$^{-1}$bb bi"s_8 i™bb. $\overset{p}{}$

Pentru ă obţine configuraţia bi"s_B i™bb şi pentru *m* =0, adăugăm şirul s_3 bs$_6$ bO.

In continuare folosim şirurile s_8*is$_7$ i* — 1. *$s_7$is$_B$* bO, $s_8$bs$_e$bl, *$s_9$is$_t$il, s_t* b.s$_{lu}$iO, •^{s}'io ^{,s}n *i* ~ .s$_n$ is$_{12}$ bO, s_{12} bs$_{13}$ b — 1, s_{13} is$_{13}$ i — 1,s_{13} bs$_{14}$ iO, în urma cărora avem calculul: bi" s_8 i™bb |— bi" ^{1} *s_7 ii^m* bb |—bi"$^{-1}$ bi'"$s_9$bb|—bi" ^{1} bi'"$^{-1}$ s_{12} bib P- bi"$^{-1}$ s_{13} bi™$^{-1}$ bib |— bi"$^{-1}$ s_{14} i™bib.

Am presupus aici *n* > 0, *m* > 0. Pentru a cuprinde şi cazul *m*

= 0, adăugăm şirurile s_8 bs_{15} iO, s_{15} is_{18} *i* — 1, s_{18} is_{14} bO.

Cu ajutorul şirurilor s_{14} is_g *iO,* s_{14} bs_6 bO, obţinem configuraţia bi" ^{1} s_6 i™ bib. Dacă n > 1, atunci această configuraţie se transformă în bi"$^{-2}$ s_8 i^m biib. Dacă n > 2, repetînd procesul, obţinem bi"$^{-3}$ s_8 i^m biiib, ş.a.m.d., pînă ce obţinem, după n aplicări ale procedeului descris, configuraţia bs_8 i^m bi" b. Transformarea acestei configuraţii se face cu ajutorul şirului s_8 is_7 *i* — 1 (dacă m > 0), sau s_8 bs_{15} iO (dacă *m* = 0), obţinînd configuraţiile s_7 bi™ bi", sau, respectiv, s_{18}bii" b.

In sfîrşit, şirurile s_7 bs_{17} bl, s_{17} is_{17} il, s_{17} bsbO şi s_{18} bs_{l8} bl, s_{18} isbO conduc respectiv la bi™sbi"b, după cum m > 0, sau *m* = 0.

Exemplul II. 13. â. Pentru dedublarea unui cuvînt de pe bandă vom folosi maşina Turing Af = (/, S, *Q* ; b, s_0), unde S şi *Q* vor fi precizaţi uite-

*

rior. Această maşină realizează calculul s_0 bi" b"$^{+3}$1— sbi" bi" bb.

-w

Şirurile s_0 b s_4 *bl,* s_L is_Y il au ca efect calculul s_0 bi" b"$^{+3}$1— bi" s_4 b"$^{+3}$ = = bi" $j bb° bb° b"$^{+1}$, unde b° = *e.*

Mulţimea de şiruri s_4 bs_2 b — 1, s_2 is_3 bO, s_3 bs_4 bl, s_4 bs_s iO, s_5 is_5 il, $s_5$$bs_B$ bl, s_B is_g *il,* s_8 bs_7 iO, s_7 is_7 i — 1, s_7 bs_8 b — 1, s_8 is_8 i — 1, transformă configuraţia bi"~* s_4 bi* bi* b în bi" <*$^{+1}$> s_8 bi*$^{+1}$ b i*$^{+1}$ (exerciţiul), în cazul în care *n* — *h* > 0.

Adăugind şirul s_8 bs_4 bO se parcurge din nou mulţimea de şiruri de mai sus, obţinîndu-se bi"$^{-}$<*$^{+2}$) s_4 bi*$^{+2}$ bi*$^{+2}$ ş.a.m.d., pînă cînd, după *n* repetări ale procedeului, se ajunge la configuraţia bs_4 bi" bi" b.

Şirul s_x bs_2 b — 1 conduce maşina la configuraţia s_2 bbi" bi" b. Conside- rînd acum şirurile obţinute din *Q (T)* prin înlocuirea stărilor {s_0 , s_x, s_2 , s_3 , > s_5 • $^{s'}$}> respectiv cu {s_2, s_6 , s_w , s_n , s_{12}, s_{13}, s_u}, ajungem la configuraţia s_{14} bi" bbi" b.

Folosim acum şirurile din *Q* (D$^+$), în care înlocuim s_0 , s_x, s prin s_{x4} , s_{15} , s_{16}, respectiv. Ajungem la configuraţia bi" s_{16} bbf'* b. Aplicînd din nou şirurile din *Q (T),* în care stările s_0 , s_4 s_5 , s le înlocuim respectiv prin

s_{16} • ^{s}i7 ’•••• ^{s}2i ’ ^{s}22 ’ obţinem bi" s_{22} bi" bb. în sfirşit, utilizînd şirurile din *Q* (D), în care stările s_0 , s_l , s sînt schimbate respectiv prin s_{22} , s_{23}, s obţinem sbi" bi" bb.

Observaţia II.13.1. Maşinile Turing considerate în exemplele precedente transformă conţinutul benzii fără a folosi celule situate la stînga primei celule a conţinutului iniţial. Acest mod de funcţionare este cunoscut în literatura de specialitate sub numele de *calcul corect.*

Observaţia II.13.2. Compunînd maşinile *D ~, D+, P,* **.W** obţinem alte ma-

«

sini Turing. Anume, maşina D$^+$ *D$^+$ realizează calculul s_0 bi" bi^m b

|— bi" bi"[V] sb. Apoi, maşina $C = P * D\sim * P$ execută transformarea bi" bi[m] s_0 bi* b]— bi* sbi" *bi"*[1] b. în sfîrşit, $K = D^+ * M *$ P* $D^+ * P' * M *$ P* $D^+ *$

P* $D * D$P * D^+ efectuează calculul s_0 bi" bi[m] |—bi" bi[m] sbi" bi[m], aşa cum se poate constata imediat.

Observaţia II.13.3. Pentru simplificarea scrierii calculelor care intră în demonstraţii, vom face următoarea convenţie de notaţie. Fie $MT = (/, S, Q ;$ b, $s_0)$ o maşină Turing cu $S = \{s_0 , s_x,..., s_{B_x}, s\}$. Vom nota cu Q_k mulţimea şirurilor care se obţin din şirurile lui *Q,* înlocuind simbolii s,- cu simbolii s_{x+x}, $j = 0, n - 1$, iar s prin s_{B+l}., iar cu $M(T)_k$ maşina (/, {s_t.....•«,+*_i *Q*; b,* $s_Ă$.).

Dacă ps$_0$ *q* /— *rsu* avem ps_k *q* ţ— rs_{n+k} *u,* ceea ce notăm prin $ps_k q(MT)_k$ MT (MT)k rs_{n+k} *u.* Corespunzător acestei notaţii, penLru compunerea de maşini avem : dacă p_0 s_0 q_0 $(U)_o$ $_{P1}$ s_n q_k (V)„ p_2 s_m q_2... (W)$_z$ $_{Pj}$ s„ atunci p_0 s_0 q_0 *{(U *V) ... « W)pjsq.*

Teorema II.13.2. *Funcţiile primare o,* **s(** şi **if** *sînt calculabile de maşini Turing.*

Demonstraţie. O maşină Turing MT_O = ({b, i}, S, *Q* ; b, s_0) calculează funcţia **o** dacă ea poate realiza calculul s_0 bi" b |— sbb"$^{+1}$, ceea ce impune prezenţa în *Q* a următoarelor şiruri s_0 bs_x bl, s_x is_x il, s_x bs_2 b — 1, s_2 is_3 bO, s_3 bs_2 b — 1, s_2 bsbO. într-adevăr, în acest caz avem :

$$s_0 \text{ bi" b} \vdash \text{bi" Sj b} \vdash \text{bi"}^{-1} s_3 \text{ bb} \vdash s_2 \text{ bb"}^{+1} \vdash \text{sbb"}^{+1}.$$

Apoi maşina Turing MT_S = ({b, i}, S, *Q ; b,* s_0) calculează s_0 bi" b |—sbi"$^{+1}$, dacă *Q* conţine următoarele şiruri : , s_0 bs_x bl, s_x bsj bl, s_x bs_2 iO, s_2 is_2 i — 1, s_2 bsbO, deoarece avem s_0 , bi" b |—

Ne vom mărgini, pentru clarilate, la cazul funcţiei i|. Avem s_0 hi" hi™ $(D^+)_o$ bi" s_2 bi" $(P)_2$ bi'" s_{22} bi" $(M\ 7'_0)_{22}$ bi"' b... b b (P ")«■''$_2$s bi'". Conchidem eă funcţia i| este calculată de maşina $D^+ * P * 3/7'_u * D$.

Teorema II.13.3. *Dacă funcţiile parţiale g ţ.i*| ,..., .r„), f_t (.i-$_{\pm}$,..., .rj,... ..., f_n (.Cj,..., .Tj.) *sini calculabile de maşini Turing, alunei funcţia obţinută prin superpoziţie f* (.tj,..., x_k) = *g* (f_j (.r, ,.... ,r_t).....f_n (.r, .r_A.)) *esle calculabilă de o maşină Turing.*

Demonstraţie. Vom considera cazul $n = k = 2$, cazul general [ralindu-se analog. Să presupunem că funcţiile *g, f_k, f_2* sînt

---

V bi" s_x b |— sbi"$^{+1}$.

calculate respectiv de maşinile turing *MTţg), MTţfJ, MTţf^.* Avem:

s_0 bi" bi^m $(A)_o$ bi" hi^m s_a hi^H bi^n (.1/ *T (ff)),,* bi" bi"^{1} sp bz^7* $(C)_3$ bi^f

$^{(“-m)}$ s_r bi" bi"^{1} (.1/7’ $(f_2))_Y$ bz^7* s_s bz^7* *(D~)$_s$* s_k bz^7* <^{n}-^{m}> bz^7* <”•’”> (.1/7’ *(g)L*

$_s$_ bi’$^(A$ *it <"."))•*

Adăugind şirul s- bsbO, obţinem configuraţia finală :

sbi»(A./.>, g

Teorema II.13.4. *Dacă funcţiile g şi h sînt calculabile de maşini Turing, atunci funcţia f* **= R** *(g, h), obţinută prin recursie primitivă, este calculabilă de o maşină Turing.*

Demonstraţie. Ne vom mărgini la considerarea recursiei primitive aplicată asupra funcţiilor *g* (,r), de o variabilă, şi *h (x, i,* z) de trei variabile, care ne furnizează funcţia de două variabile *f* definită prin: *f(x,Q) = g (x), f (x, i* + 1) *= h (.?, i, f (x, i)).*

Notăm *h (x, i, f (x, i)) = k (f (x, i), x).* Fie *M T (g)* şi *MT(k)* maşinile Turing care calculează funcţiile *g* şi *k,* respectiv. Avem *s_0* bi” bi"^{1} b $(D^+K$ *P* *.1/ » *P* * $D^+)_o$ bi” bi”^{1} s_a bi” b (.1/ *T (gf)$_a$ bi" bi™ Sp* bz^7<"*®>.

Introducînd acum comenzile s_3 bs_{p+1} b — 1, Sp_{+1} isp_{+2} bO, configuraţia devine: bi” bi”$^{1'1}$ s_{B+2} bbz^7<$^{n><,}$>. Apoi avem bi" bi”$^{1-1}$ sp_{+2} bb $z^{7(}$".°) (7’)p_{+2} bi" bi"$^{1-1}$ s_Y *bP$^{(n}$ °>* b *(D~ * P * D$^+$ * P * M)$_r$ bi’”’^{11}* bi$^{/(n}$-°) sj bi" bi” b *(P* D * *P « D$^+$)$_s$* bi” bi"$^{1'1}$ *s_z* biA"-») bi” b *(M T (k))$_e$* bi” $bi^{1"-1}$ s_T b i^”-») b.

Folosind comanda ,s_T bsp bO calculul se reia de la prima apariţie a stării Sp, realizîndu-se transformarea configuraţiilor de forma bi" *bi^{m}~)* Sp $bz^{7("7)}$b în configuraţii de forma bi" $bz^{"1}$~<^{1} $^{+ 1)}$ s_T $bi^{/("7+1}$> b atîta timp cît *n > j,* ajun- gîndu-se în cele din urmă la configuraţia bi" bsp biA"*”$^{1)}$ b, care trece în : bi” s_{3+1} b bi^"’^{1}”) b. Folosind şirul s_{3+1} bs_{T+1} bO, obţinem bi” s_{T+1} b $bz^{7("'m}$)b, care se transformă astfel :

bi" s_{T+1} bbz^7<$^{n-m}$> b *(T),$_{+\ 1}$* bi” s_x b z^7("*”O *bb(P *MT$_0$* * P)$_x$* bs_g bi*””^{1}) b (D')^ s_v bbz^7<”-"» b *(T„)Su* bF<”-’") b.

Un ultim şir bsbO ne conduce la configuraţia sbi^"-”>b. ■

Observaţia II.13.4. Ani demonstrat pînă acum că orice funcţie primitiv- recursivă este calculabilă de o maşină Turing.

Teorema 11.13.5. *Dacă funcţia parţială f* (.r, ,..., rr_m, y) *esle*

calculabilă de maşina Turing .V *T* (f), *alunei funcţia g (*x_t ,..., r,n) = *p.*$_y$ *(f* (J-J ...., x_m. *y)* = 0) *esle calculabilă de o maşină Turing.*

Demonstraţie. Vom da demonstraţia pentru cazul *m* = 1. Pentru orice n s JV avem .s_0 bi" b (K)$_o$ bi" bs$_a$ bi" b *(M T (f))$_a$* bi" bi° sp J»rZ(».o).

Mulţimea de şiruri Sp b.s-p$_{+1}$ bl, s-p$_{+1}$ i.s'p$_{+2}$ bO, .Sp$_{+2}$ b .Sp$_{+3}$ b — 1, s$_{e+3}$ b.Sp$_{+4}$i(), .s-p$_{+4}$ *isp*$_{+5}$ *i* 1 transformă configuraţia bi" bi* bi$^{/(I}$-** în configuraţia bi" bi*$^{+1}$ s$_{p+5}$ bi/(».O-i.

înlr-adevăr, avem succesiv : bi" bi* sp bi$^{/(n,}$b |— bi" bi* b.sp$_{+1}$ F<"-*) [— bi" bi* b.s-p$_{+2}$ bi*$^{(''}$'*>$_{-}$ ^{1} |— bi" bi* .s-p$_{+3}$ bbi$^{z<}$"-*>~^{1}1— bi" bi* ,s'p$_{+4}$ ibi*$^{(''}$'*> $_{-}$^{1}1— bi" bi*$^{+1}$ .s'p$_{+5}$ bi/*"-*)$^{-1}$, calcul care se poate executa atîta timp cit f(u,y)> 0.

în continuare avem bi" bi*$^{+1}$ .sp$_{+5}$bi$^{z(}$"-**$^{-1}$ *(M*T$_o$)p$_{+5}$ bi" bi*$^{+1}$ *s$_r$ b ... b (D * * D~)$_r$s$_s$* b *i^n bi*$^{+1}$* (/<), bi" bi*$^{+1}$.s$_e$ bi" bi*$^{+1}$ *(MT (f)| bi» bi*$^{+1}$* bi*("-* $^{!}$ D.

Folosind şirul .sb.sbO, ajungem la configuraţia bi" bi*$^{+1}$ bi*l"-* $^{!}$». Calculul este reluat, obţinîndu-se în continuare bi” bi*$^{+2}$ Sp bi$^{Z(",*+2}$> ş.a.m.d.. atîta timp cît *f (n,j)* > 0.

Dacă la un număr dat avem *f (n, j)* = 0, obţinem configuraţia bi" bi*.s-pbi° = = bi" bi* .s'p b, care se transformă în bi" bi* bsp$_{+}$i b.

Folosind şirul sp$_{+}$i b,s$_{x+1}$ bO, obţinem configuraţia bi" bi* bs$_{T+2}$ b, care se transformă astfel : bi" bi* bs$_{T+}$i b(b~ * D“)$_{T+1}$bi” s$_e$bi* *(P* MT$_0$*D~')$_6$s^]ti^1.* în sfîrşit, şirul. s$_u$ b.sbO ne conduce la configuraţia s'bi'$^{(''}$>.

Este clar că valoarea lui *j* pentru care s-a obţinut configuraţia bi* este primul număr natural, pantru care *f (n,j)* = 0 şi astfel *g* (n) = *j.*

Dacă în procesul de calcul *f (n,j)* > 0, Vj s *N* atunci maşina Turing nu se opreşte niciodată, dar în acest caz nici valoarea *g* (n) = *ii$_y$ (f (n,j)* = 0) nu-i definită. Conchidem astfel că maşina Turing construită mai sus calculează funcţia *g.* ■

Teorema II.13.6. *Orice funcţie parţial recursivă esle calculabilă de o maşină Turing.*

Demonstraţie. Rezultă din teoremele II.13.2 —5. ■

Observaţia II.13.5. Fie *E* o mulţime recursivă *E* c /*. (v. definiţia 1.7.11). Atunci funcţia ei caracteristică y_p este o funcţie recursivă şi de aceea funcţia y_E *: I* -+ I** dată de :

$$7.E\ (/^*) \quad \begin{cases} i, \text{ dacă } p \text{ G } E, \\ \text{nedefinită, în caz contrar,} \end{cases}$$

unde *i* este un simbol fixat din *I* este o funcţie parţial recursivă.

în baza teoremei II.13.6, această funcţie parţial recursivă este calculabilă de o maşină Turing *MT = (I, S, Q;* b, s$_0$). La introducerea unui cuvînt *p* maşina se opreşte (adică trece în

starea finală) atunci şi numai atunci cînd *p* G *E,* ceea ce arată că limbajul *E* este recunoscut de maşină prin starea finală.

Prin urmare, mulţimile recursivc din *1** sînt limbaje dc tip 0, aşa cum rezultă din teorema 11.12.4.

Precizarea raportului dintre aceste două clase de limbaje va fi făcută în paragraful următor.

§ 11.14. MULŢIMI RECURSIV ENUMERABILE ŞI LIMBAJE

Teorema 11.14.1. *Orice limbaj de. tip* 0 *este o mulţime recursiv enumerabilă.*

Demonstraţie. Fie *L* , deci *E* = L (G), unde *G = (*I_N , I_T , x_Q ; F) este o gramatică de tip 0. Pentru orice *m e N** există numai un număr finit de deducţii D_m = {x_0 => *p* | *p* /*}, în care se folosesc exact *m* reguli de generare din *F.* Astfel există un algoritm în baza căruia putem alcătui lista deducţiilor (J *{*D_m|*msN*}.* Din această listă vom reţine sublista conţi- nînd deducţiile în care *p* G I'_T . Astfel putem alcătui o listă cu cuvintele din *L* (G), care este deci recursiv enumerabilă (v. observaţia 1.7.16). ■

Teorema II.14.2. *Orice limbaj de tip* 1 *esle o mulţime recursivă.*

Demonstraţie. Este suficient să arătăm că pentru un limbaj *E* de tip 1 există un algoritm, care ne permite să decidem dacă un cuvînt *p* aparţine sau nu limbajului, deoarece în acest caz există un algoritm care calculează valorile funcţiei caracteristice XE • Astfel, în baza tezei lui Church, funcţia y_E va fi o funcţie parţial recursivă.

Fie deci *E = L* (G), unde G este gramatica dc tip 1, G = *(*I_N, I_T , x_0 ; *F).* Avem *e* G *E,* dacă şi numai dacă (.r_0 -> *e)* s *F* deci, putem presupune *p e.* Să considerăm şirul de cuvinte :

(11.14.1) $p_0 = x_0$, p_Y ,..., p„_i , *p„ = p,* unde *n* > 1, p_j s *(*I_N [J I_T*)*,* pentru *j / k* avem p_j / p_k şi *l (pj) < l* (p, + i) pentru *j* = 0, *n* — î. Este clar că pentru un anumit *p* mulţimea acestor şiruri este finită şi *p G L (G)* dacă şi numai dacă există un şir (II.14.1) astfel îneît:

(11.14.2) = p_0 => Pi => ... => p_n_i => p_n G .

Prin urmare, pentru a decide dacă *p* G *L* (G) este suficient să

căutăm în mulţimea finită a şirurilor (II.14.1) un şir de forma (II.14.2).

Rămîne să observăm că pentru orice două cuvinte $, p_{j+}i\ e\ I_T)^*$ se poate decide într-un număr finit de paşi, dacă sau nu $p_t => p_{j+1}$, deoarece F este o mulţime finită. ■

Rezultatul următor arată că familia esle strict inclusă în clasa mulţimilor recursive.

Teorema 11.14.3. *Există un limbaj recursiv care nu este limbaj de tip* 1. *Demonstraţie.* Ne vom fixa la un alfabet terminal cu doi simboli.

Fie gramatica de tip 1, $G = (I_N$, {R , i_2}, x_0 ; *F).* Deoarece limbajul generat de gramatică nu este afectat de modul de notare a variabilelor neterminale. vom considera aceste variabile ca o parte finită a mulţimii numă- rabile {.r I *j e* _V}. Să considerăm acum alfabetul $I_T = I_T$ tj {->, !}.

Dacă *F = [/■; -» pj| j =* 1,/}, mulţimea/[7], deci gramatica *G* poale fi repre- zenlată de cuvinlul : r_L ->■ p_L 1 p_2 -> q_2 ! ... ! p_1 -> q_{lt} din alfabetul Jy .

Să considerăm homomorfismul o:(/T|J /A)* -» *1*T* definit pe

φ '2 '1 '2)*2 *1 '2 *2*1 ᶜ2

generatorii semigrupului liber *(IT* |J *Ix)** prin :

'1

Astfel, dacă, de pildă. *F* = {.r_0 -> *e, x0* -» .Tj .r_2, .T_2 -»i\ , x_l x_2 ->■ i_2 .r_2} atunci /'' este reprezentat de cuvîntul *p* (G) = 9 (.T_0 -» *e* 1 x_0 ->■ $x_±$ x_2 ! x_2 -> i_x ! x_1 x_2 -> -» i_2 x_2) = i_2 î? i_2 l'$_2$ *1 *2 *2 *2 *2 *2 *2 '1 '2 f2 VIl '2 *i'2 *1* '2 [2 *î ?2 ,#2 'î f2 f2 '1 f2 *2 *1 f2 i_2 *i* i_2 i_2 iţ i_2 i_2 i_2 i_2 i® i_2 i_2 i_2 i_2 iţ i_2 •

Conchidem că oricărei gramatici de acest tip îi corespunde un cuvînt din alfabetul /y . Reciproc, avînd un cuvînt din I_T , putem decide dacă el corespunde sau nu unei gramatici de tip 1.

Mulţimea *l'T* este numărabilă deoarece elementele ei, cuvinte de lungime finită, pot fi puse în corespondenţă biunivocă cu o parte a mulţimii numerelor naturale scrise în sistemul ternar ataşat literelor $i_±$ şi i_2 respectiv cifrele 1 şi 2. Astfel cuvîntul ij i_2 <\ i_2 î$_2$ se reprezintă ca numărul 12122 etc. Vom putea numerota deci elementele din *I?* . Procedăm inductiv, după lungimea cuvintelor. Cuvîntului vid îi atribuim numărul 1.

Să presupunem că mulţimea *{p | p* s *i*T , l* (p) *k}* a fost deja numerotată şi că | {p | p s *l'T , l* (p) A} | = *m.* Atunci celor 2^{l+1} cuvinte de lungime A 1 li se atribuie în ordine lexicografică[1]) numerele m-f-1,..., *m* + 2^{A+1}.

VI) **Cuvîntul** *p* **precede cuvîntul** *q* **in _ordine lexicografică dacă** *p = riLu,* **iar** *q* **= ri$_2$u.**

Numerotarea cuvintelor din I^*_T induce o numerotare a gramaticilor de tip 1 : gramatica G_j corespunde celui de-al j-lea cuvînt din F_T , care defineşte o gramatică de tip 1. Această numerotare a gramaticilor este făcută cu repetiţii, deoarece putem avea $Gj = G_t$ cu j / k, deoarece schimbînd ordinea regulilor de generare, obţinem cuvintele p (Gy), p (G_t) distincte.

Fie $L = \{p_y \mid pj \pounds L (Gy)\}$. Avem e *L* deoarece $e = p_v$ G *L* (G_x) căci ^1 = ({^o}' » -To > {^-o "* e})-

Afirmăm că limbajul *L* este recursiv, adică funcţia lui caracteristică y_L este o funcţie recursivă. Conform tezei lui Church, rămîne de arătat existenţa unui algoritm de calcul al acestei funcţii, ceea ce revine la existenţa unui algoritm, care pentru orice p J_T este capabil să rezolve problema apartenenţei lui p la L.

Ï-'ie aşadar p G]*t . Atunci există j G A', astfel incit $p = p_j$. Cum gramatica G_f este de tip 1, în baza teoremei II.14.2, există un algoritm de decidere a apartenenţei lui $p = p_f$ la *L.*

Limbajul *I,* nu este de tip 1, deoarece dacă *L* G , atunci ar exista *j* G A\ astfel ca *L* = 7, (G_j). Fie al j-lea cuvînt *pj.* Dacă *p e L = L* (G_j), atunci $p = pj$ *L* (G_j), ceea ce conduce la o contradicţie. Dacă p_j *L (Gj) =*

= L, atunci $p = p_j$ G *L* şi avem o nouă contradicţie. Astfel, presupunerea e $j)_Y$ este absurdă. ■

în continuare vom arăta că se poate numerota mulţimea maşinilor Turing cu alfabetul de intrare *I, / 1*1 < oo , ceea ce ne va permite să arătăm în final că familia limbajelor de tip 0 pe alfabetul *I* este strict inclusă în W

Putem presupune totdeauna că mulţimea stărilor maşinilor Turing considerate este de forma $\{s_0$, *,..., s_m* fără a afecta calculul efectuat de maşină sau limbajul recunoscut de ea. Fie *1* **= {b,** ,..., i_{o_t}}.

Definiţia II. 14.1. *Gradul unei maşini Turing* este numărul natural gr *(XIT) =* = max {m, n}, unde *MT* **= ({b,** i_1,..., i_n .,}, (s_0 , s_L,..., s^j}, *Q* **; b ,** s_0).

Teorema II.14.4. *Mulţimea maşinilor Turing cu alfabetul de intrare I,| 1*1 <oo *este numărabilă.*

Demonstraţie. Să observăm că există $3m^2$ n^2, 5-uple de forma *sis'i'a.,* uude *s, s'* s {s_0 ,..., i, *i'* G {b, ij,..., i.^J, a G { —1, O, 1}.

Mulţimea = *{M 7* | gr *(XIT)* = A-} este finită, deoarece cu $3m^2$ n^2 5-uple $3m^2$ n^2 <31^ se poate forma un număr finit de mulţimi *Q* distincte, şi fiecare mulţime *Q* determină în mod unic o maşină Turing şi reciproc. Astfel mulţimea 77 a maşinilor Turing este numărabilă, fiind o reuniune numărabilă de mulţimi finite.

Teorema II.14.5. *Mulţimea părţilor recursiv enumerabile*

din I, £ (/*) este numărabilă.*

Demonstraţie. într-adevăr, dacă *E* este o mulţime recursiv enumerabilă, există o funcţie parţial recursivă $f_B : D \to N$, *D* CZ *N,* astfel înеît $E = \{f_E(n) \mid I\ n\ e\ D\}$. Rezultă de aici că există o aplicaţie surjectivă <p: *£ (I*)* -> 77, unde ?(E) = *XIT* (f_E). ■

Consecinţa II.14.1. Există părţi ale lui *I** care nu sînt limbaje de tip 0. deoarece mulţimea *<p(I*)* nu-i numărabilă, în timp ce familia părţilor recursiv enumerabile pe acelaşi alfabet (deci, în particular şi familia limbajelor de tip O pe 7) este numărabilă.

Se ştie că familia mulţimilor recursive este inclusă în familia mulţimilor recursiv enumerabile. Această incluziune este strictă, aşa cum rezultă din

Teorema II.14.6. *Există o mulţime recursiv enumerabilă, care nu este recurşi vă.*

Demonstraţie. în baza teoremei 1.7.1, esle suficient să construim o mulţime *H* recursiv enumerabilă, a cărei complementară H^e nu-i recursiv enumerabilă.

Fie $I = \{<\}$ un alfabet cu o singură literă. Dacă *E* c *I** este un limbaj recursiv enumerabil, atunci $E = \{/'(,i)|ne N\}$, unde *f* este o funcţie recursivă. Există atunci un algoritm de calcul al funcţiei *f,* algoritm care se poale descrie printr-un text obişnuit, care consitituie un cuvînt (în sensul adoptat în această lucrare) asupra unui alfabet *I'* (de exemplu *I'* = {a,..., *z,* 0,...9}.). Cum mulţimea limbajelor recursiv enumerabile este numărabilă, mulţimea cuvintelor din /'* care descriu algoritmii corespunzători este numărabilă. Fie ea $[p_1,..., p_n, ...}$ şi fie f_n funcţia calculată de algoritmul descris de cuvîntul p_n .

Definim un limbaj *H* s /* astfel :

$$H = \{i» \mid n \in N, 3Ă-, i» = f_{n+k}(k)\}.$$

Limbajul *H* este recursiv enumerabil. Pentru a arăta aceasta, să notăm cu *(k,j)* pasul *k* din algoritmul descris de cuvîntul p_j . Cum orice algoritm arc un număr finit de paşi , facem convenţia că, dacă *k> r.j,* atunci în pasul *(k,j)* algoritmul nu are nici o acţiune.

Vom executa paşii diferiţilor algoritmi în ordinea :

(1, 1), (1, 2), (2, 1), (1, 3), (2. 2), (3, 1), ele.

în această ordine locul *h* ocupat de perechea *(k. j)* esle dat de

$$h = \tau(k, j) = \frac{(k + j - 2)(k + j - 1)}{2} + k$$

Aplicaţia *x : N* x iV -> *N* este bijectivă (v. exemplul 1.1.11). Să considerăm acum funcţia *f = D -> I*, D £ I** dată de :

$$/■(i») \quad \begin{cases} P \text{ , dacă } i^j \text{ este obţinut în pasul } (KJ_{+1}, j + 1) \text{ şi } n = T\ ,j + 1) \\ \text{nedefinit, în caz contrar.} \end{cases}$$

Mulţimea valorilor funcţiei parţial recursive *f* este chiar mulţimea *H,* prin urmare *H* este o mulţime recursiv enumerabilă.

Limbajul H^e nu este recursiv enumerabil. Dacă H^c ar fi recursiv enumerabil, atunci ar exista un număr *k,* astfel ca H^e să fie generat de algoritmul descris de cuvîntul p_{k+1}. Pentru orice n > 0, cuvîntul i" H^c dacă şi numai dacă el nu este generat de algoritmul descris de cuvîntul p_{o+1}. Prin urinare, i^k H^c dacă şi numai dacă el nu esle general de algoritmul descris de cuvîntul p_{t+1} , ceea ce contrazice presupunerea făcută. ■

§ II. 15. LIMBAJE DE TIP 0

Vom arăta că maşinile Turing au faţă de limbaje de lip O rolul de dispozitive de recunoaştere.

Teorema II.15.1. *Un limbaj de lip* 0 *esle recunoscui ele o maşină Turing. Demonstraţie.* Fie G = *(I_N , I_T , x_u ; F)* o gramatică de tip 0 şi *L = L (G).* Vom construi o *M T* nedelerministă care va accepta limbajul *L.*

Fie *MT = (I_N* IJ *I_T* U {•», *. x}, S, *f*; b, $s_{()}$), unde {b, #, ,r} Q *(I_N* |J *I_T) =* = 0 cu mulţimea de stări finale *$S_{\pm}$ S, f* fiind precizat ulterior.

Vom înscrie iniţial pe banda maşinii un cuvînt *p* s *F_T* . într-o primă fază *MTNĎ* adaugă pe bandă înainte de cuvîntul *p* simbolul #, iar după cuvîntul *p* şirul # r_0 #. Apoi, *MTND* fixează banda în aşa fel, îneît x_0 să se afle în dreptul capului de citire şi se începe un proces de simulare a deducţiilor din gramatica G. Maşina are acum posibilitatea de a înscrie în locul lui ,r_0 , între simbolii #, oricare din cuvintele *i| ,..., r_n,* unde *(x_0 ->■ r_j)* e *e F, j* = 1, n. Apoi ea alege in mod nedeterminist o pereche de numere (/i, t) cu 1 *< t <* max {/ (p) j (p -» *q)* G F}. Dacă cuvîntul *r* înscris în locul lui *r_0* conţine, începînd de la poziţia *h,* un subcuvînt de lungime *t* care este partea stingă a vreunei reguli de generare din F, adică *r = r'r_g r"* cu *l* (r') = = *h* — 1, *l* (r_0) = *t* şi mulţimea {<p | (r_0 -> *qf)* G F} 0, maşina are posibilitatea de a înlocui cuvîntul *r_0* cu oricare din cuvintele *q_j,* deplasînd la dreapta sau la stînga cuvîntul *r"* # pentru ca inserţia lui *q_j* să se facă fără spaţii libere. Acest mod dc a acţiona sc continuă pînă cînd inserarea nu mai este posibilă.

Dacă ultimul cuvînt este format numai din simboli terminali, se compară acest cuvînt, simbol cu simbol cu p, cuvînt scris iniţial pe bandă. Dacă cele două cuvinte coincid, cuvîntul p este acceptat. Ca atare, un cuvînt p este

recunoscut de *MTND,* dacă şi numai dacă $r_0 \overset{*}{\Rightarrow} p$, adică p e *L (G).* ■

Afirmaţia din enunţ rezultă ţinînd seama şi de teorema II.12.3. ■

Teorema II.15.2. *Un limbaj este de tip* 0, *dacă şi numai dacă este acceptat de o maşină Turing printr-o mulţime de stări finale.*

Demonstraţie. Faptul rezultă din teorema precedentă şi din teorema II. 12.4. ■

Teorema II.15.3. *Familia de limbaje este strict inclusă in familia de limbaje J>$_0$, adică*

Demonstraţie. Sc ştie că orice mulţime recursivă din *I** este un limbaj de tip 0 (v. observaţia II.13.5), iar teorema II.14.3 asigură că există un limbaj recursiv care nu este un limbaj de tip

1. ■

Am arătat în § II.2 că limbajele de tip 0 sînt închise Ia operaţiile de reuniune, produs şi închidere prin produs (v. teorema II.2.7).

Vom demonstra acum :

Teorema II.15.4. *Intersecţia a două limbaje de tip* 0 *esle un limbaj de lip* 0.

Demonstraţie. Fie *G' =* (fj,, *I'T, x„ ; F')* şi G" = (7jj, 7j!, xj'; F") două gramatici de tip 0 şi *L' = L (&)* şi *L" = L* (G") limbajele de tip 0, generate de aceste gramatici respectiv.

Putem presupune, fără a restrîngc generalitatea că *I'N* Q 7jj *= 0.*

Să considerăm gramatica :

G = &U W 4U - [x]o i^'U [F]"U [F]). unde I_N conţine pentru fiecare simbol *i' <= I'T0 I'T* un element $X_($ şi în plus elementele X_Q , X_x, X_2, X_3 , iar *F* constă din următoarele reguli dc generare :

(II.15.1) X_o -» Xj X_Q X_2 X_3 X_Q XJ

(11.15.2) X_2 X_3 i -> $X_($ X_3, pentru orice i s *I'T* (J

(11.15.3) *1"* i_x $X_($ -» X, i_x, pentru toţi i, i_x *e I'T* (J

(11.15.4) Xj *Xt i -* iXj* X_2, pentru orice i e *I'T* (J

(11.15.5) *I'T*

X, i „ iX„ pentru orice i e *I'* (J 7i !

(II.15.6) Xj X_2 X_3 Xj -> *e.*

Prima regulă de generare care apare înlr-o deducţie X_Q => *p* este regula

(11.15.1).

în continuare, aplicînd regulile de generare din *F'* şi *F",* obţinem cuvîntul *r* = X_1p'X_2X_3p"X_1 cu *p'* s *L'* şi *p"* e *L".*

Simbolii X, , X_2, X_3 pot fi eliminaţi numai prin folosirea regulii (II.15.6). în acest scop apare necesară folosirea regulilor (II. 15.2 —5).

Aplicarea acestora se poate face, cum se constată uşor, dacă şi numai dacă *p' = p"* e *L'* Q *L"* şi astfel obţinem cuvîntul *p' XL X2 X3 Xr* care, folosind regula (II.15.G), Lrece în //, deci *L (G) = L (G')* Q *L (G").*

într-adevăr, fie *p'* = ij ij ... *i'm , p"* = ij' ij' ... *i„* ■ Avem succesiv X_o => Xj ij i_2... i_m X_2 X_3 ij i_2 ... $i_„$ Xj => X_x i_1 ... i_m X/' X_3 i_2 ... $i_„$ Xj =>

. (II.15.2) (II.15.3)

=> Xj $X_($" ij... 4 X_3 ij'... ij'X_x => ij Xj X_2ij... 4 X_3ij'... ij, Xj aplicarea (11.15.3) (II.15.4)

regulii (II.14.4) putîndu-se face dacă şi numai dacă ij = ij' .

In continuare avem: ij Xj X_2 ij... *i'm* X_3 ij' ... *i" X1* => ij Xj ij... ij,

X_2 (11.15.5) .
X_3 ij'... i'j X_t şi procedînd la fel, găsim ij' = ij şi obţinem : X,, => ij ij Xj ij... ij, X_2 X_3 ij'... ij'Xj, etc. Este evident că avem $m = n$ şi $p' = p''$.

Teorema 11.15.5. Fie E , E s I_T şi u, v I_T . *Există limbajul* E_t e , *astfel Incit:*

') Fi = {*P* I *P* = *lli* "7- *J* > 7 e F}-

ii) V</ s *E există j* s *X* astfel ca u* i> q* E_L.

Demonstraţie. Fie *li = L* (G) cu $G = (I_N , I_T , x_0 ; F)$. Să considerăm gramatica $G_r = (I_N$ |J , z;}, I_T (J {zz, zz} , r', (J F_2), unde *x'o , y & I_K ,* iar F_t = {/■ -» *iv | (/• -> iv) e F, l* (z) < *l (iv)}* |J {/• -> *iv | (/• ->■ tv)* e *F*
şi *l* (r) > / (zzz)} şi *F.,* = {.r_0 -+ vx_0 ; *yv* -> zzzz} |J *{ly -* yt* I *l* e J I_T |J {/>] '.

Această gramatică este expansivă (v. definiţia II.10.1) şi prin urmare, limbajul 7, (GJ este de tip 1 (v. teorema II.10.2).

Avem E_t=*L* (GJ, deoarece dacă x_0 => q_m atunci x_0 => vx_0 , => vy^m *q=>y vy^m q G , G1 G1 Gi => yv y^{m-1} q => uyvy^m~² q => u² vy^m~² q => u^mv q* şi evident E_t satisface condiţiile z) şi *ii).*

TeoremaII. 15.6. *Pentru orice limbaj de tip* 0, *E* c *I*_T există un limbaj de tip* 1, *E.* c /* ,uzi *homomorfism* o: (/$_T$ (J *tu,v})*D, *si un limbai regulat K* e *(Q (I'_T), astfel incit*

(II. 15.7) $E = _?(E_x) = K$ | E\,

unde am notat K | E_y = {r | 3p s *K, q* e E\, *pr* = ţ}.

Demonstraţie. Fie E_1 limbajul definit în demonstrarea teoremei precedente, iar <p homomorfismul definit prin :

? («) = <P 00 = «,

cp (z) = z, Vi e I_T.

Avem cvidenL egalităţile (II.15.7) luînd *K* = {zz^m *v | m* e A'}. g

Observaţia 11.15.1. Homomorfismul cp şterge simbolii zz şi *v* din cuvintele seinigrupului liber *(I_T* (J {u, zz}) şi cum un homomorfism este o operaţie de substituţie, se justifică observaţia II.10.6, substituţia cp fiind singulară.

Observaţia 11.15.2. Egalităţile (II.15.7) arată că clasa limbajelor de tip 1 nu-i închisă la homomorfisme oarecare şi la operaţia „ | „ cu limbaje regulate, limbajul deţinut putînd fi în *Jţ₀.*

Teorema 11.15.7. *Familia de limbaje* J?$_0$ *este inclusă la*

operaţia de oglindire, adică, dacă E J>₀, atunci E e J>₀.

Demonstraţie. Fie G = *(I_N , I_T , x₀ ; F)* o gramatică de tip 0 şi *E = L (G).* Să considerăm gramatica G = *(I_K , I_T , x₀ , F),* unde *F* = {z- -> *q | (/■ -» q)* e F}. Este clar că G este o gramatică de tip 0. Avem *L (G)* = Ii. Vom arăta

că $x_0 \Rightarrow r$, dacă şi numai dacă $x_0 \overset{*}{\Rightarrow} r$ procedînd prin inducţie după lungimea $\overset{\bullet}{6}$ $\overset{G}{.}$ *n* a deducţiei. Pentru zi = 1 afirmaţia decurge imediat din definiţia lui *F.*

Să presupunem că ea arc loc pentru *n* şi fie deducţia $x_0 \Rightarrow i| \Rightarrow r$ de lungime $_{G\,G}$ *n* + 1. Atunci, conform ipotezei inductive, avem $x_g \Rightarrow i|$ şi în baza defini- ţiei lui *F,* $i| \Rightarrow r$, de unde .$r_0 \Rightarrow r$ şi reciproc. ■

$_{G}$ $_{G}$

Observaţia II. 15.3. Argumentaţia de mai sus este valabilă şi pentru familiile de limbaje^! şi J?$_2$. Conchidem că şi aceste familii sînt închise la operaţia de oglindire. în ceea ce priveşte familia această proprietate a fost stabilită în teorema II.4.12.

Teorema II.15.8. *Familia esle închisă la operaţia de substituţie.*

Demonstraţie. Fie $G = (I_N , I_T , x_0 ; F)$ o gramatică de tip 0, I_T = {p,... ..., z„} şi $E = L (G)$.

Să considerăm operaţia de substituţie <t>$_v$ generată de familia de limbaje M = {^i E_n}, *Ej* **e** *J*>$_0$, *Ej* = *L* (Gj), unde $G_s = (I_Nj, I_{Tj}, x_{0J} ; Fj),j =$ = 1, *n.*

Fără a restrînge generalitatea, putem presupune că alfabetele neterminale sînt disjuncte două cîte două şi că nici un simbol terminal nu apare în partea stingă a vreunei reguli de generare.

Limbajul *(E)* va fi generat de gramatica $G' = (I'_N , I'_T , y_0 ; F')$, unde

I_N = U U U {l/o ' {/' $^{Z}1$ '■••> ^{Z}n }' — U ^*Ti* ' *^{ar} £ ' = E^1 U F^2 U U E^3 U U *E,* U K >^{a}*y* ^{e}}< ^{un}de E^1 este mulţimea regulilor de generare >=i

n J-l :=1

obţinute din regulile de generare ale lui *F,* înlocuind în fiecare regulă simbolul i_j prin z,- , *j* = 1, *n,* F^2 = {yz_j -* yx_{oj} / *j* = 1, n} şi F^3 = {*yi* -> *iy* / i e I_{T1} tj ... tj I_{Tn}}.

Vom arăta că (E) = *L (G').* F'ie *p* s <b$_M$ *(E).* Există atunci *q* e *E* astfel încît *p* se obţine din *q* prin înlocuirea fiecărui simbol terminal i_s printr-un cuvînt *Pj* din limbajul E_j . Să notăm cu q_z cuvîntul obţinut din *q* prin înlocuirea fiecărui simbol terminal i_j prin z_f , *j* = 1, *n.*

în gramatica *G'* avem deducţia :

!/o $y^x o$ =* ytfz $y^z n$ ■■■ ^{z}*im* =* ^{Z}*iz* ^{z}*îm ypj*$_x$ ^{z}*j*$_2$ ■■■ ^{z}*J*$_m$ =* *p*^ $y^z i_2$ —

$^{Z}i_m$ ^{P}*il* •" ^{Z}*im Pil P Pi-2.* ^{Z}*i-3* — $^{Z}i_m$ *Pil Piz* — Pi_m *U* =* *Pii Piz* Pi_m = *P* şi deci *p* e *L (G').*

Incluziunea inversă se arată folosind definiţia mulţimii *F'.* ■ Imaginile prin homomorfism ale intersecţiilor de limbaje de tip 2 conduc la obţinerea tuturor limbajelor de tip 0. Are loc :

Teorema II.15.9. *Orice limbaj* E e *poate fi exprimat sub forma E =* cp (7ij P| E$_2$), *unde* <p *este un homomorfism, iar* E_1, E_2

Demonstraţie. Să presupunem că *E* = este generat de gramatica de tip 0, $G = (I_N, I_T, x_0; F)$, unde F = {r$_($ -»<7, I *j* = 1, *k}.* Să notăm *I* = = I_N *U* I_T Ş> f>e alfabetele I_T- = {i' | *i* s I_T], I_{yz} = $_{Zj}$ $_{Z|t}$}.

Să notăm cu *p'* cuvîntul obţinut din *p* s *I'T* prin înlocuirea fiecărui simbol *i* prin *i'*. Prin convenţie *e' = e.* Fie alfabetul :

Ii = IN U *IT* U *IT* U *IyZ* 9* ? : *Ii I*T* homomorfismul definit prin :

$$(u) = \begin{cases} i, \text{ dacă } u = i' \ e \ I_T \\ e, \text{ dacă } u \ e \ I_N \text{ (J } I_T \text{ U } I_v z- \end{cases}$$

Să considerăm limbajele :

#i = (Pi *Pi* z *j* r *z 'J*r *2 <li*r *iy* I z i > *'2* e *V' j* = 1, *k}** | {e}) (4)*

E2 = {x0 *z,y|J* = 1, *k} {i|r2 y rt zh r2 y|rlt r2e FT, h* = 1, *k}*{pyp', pepT }.* Aceste limbaje sînt de tip 2. într-adevăr *ET* este de forma *Er = (H*|* {e}) . • *(I'T)** Şi va deci suficient să verificăm că *H* este un limbaj de tip 2.

Limbajul *H* este generat de gramatica de tip 2, *G"* = ({cn , *i|, v2}, , D0 ;* (a0 *vt y, v1 -» TV1 x, -> p} z, v2 q}, v2-> xv2 x, v2 -» y•* | *x* G *I, j* = 1, *k},* aşa cum se constată imediat. La fel se arată că limbajul *E2* este de tip 2.

Să arătăm că *E* = <p *(E1* Q *E2~).* Fie *p* G *E.* Deoarece *E = L* (G) există un *m* G *N,* cuvintele *rf , r}* G *I** şi indicii *g(j)* cu *g* (j) = 1, *k* definit pentru *j* s 0, m astfel încît :

r*jl 9g(j) ri2* = r G+l) *Pg(i+1)* z (/+l)2 ' unde /oi = z o2 = e > *Pg(q)* ~ ^0 şi p*ml 9g(m) 'm2* = *p,* adică cuvîntul *p* este generat prin deducţia :

.T0 = r01 p p(o) r02 => r01

«

?»(0) '02 - r*ll* (1) z12 r*ll* 7 3 (1) z12 - z21 *Pg* (2) '22 T- ' *(m* -1) 1 2 *'(m-1) 2* ~

'ml Pg(m) 'm2 'ml 9g(m) 'm2 P'

Să considerăm aplicaţia 6:7* x {1,..., 7c} x 7*-> 7} dată prin 6;(/J,J, r2) = *'i Pj* z *j* r *2 y~*r *2* O *9* Şi rie cuvîntul r = 6 (r01, p (o), 7O2). 6 (rn, c? (1). *>12)* - 6('mi, *9* (m)' r *m2) P' >* avem *r* s *E±.*

Fie aplicaţia a : 7* x {1,..., />} x 7* -> 7} dală de o (/j *,j, r2)* - *r± r2 y zjr2y.* Putem scrie *r = x0* zg(o) *y* r01 *q^0)_r02_y* /n pp(1) *zt(1}* r^y^ ...

r*(m—1)1 9g(.m-l)*+(m-l)2 1/+ml *Pg(m) ~0<m) ' m2 9-* 1 *ml 9g(m)* r*m2 9* ~^0 *?(0) 17 '11 Pg(l')* 112

9 r*ll Pgll)* z*g(l) '12 9 ••• 'ml Pg(m) ' m2 9 'ml Pg(m)* z*g(m) Gn2 9 'ml Qg^m) 'm2 9P* = a o *Zg*(o) ?'(z-iiP,(i) , ?(1). zi2)- <r(' ml Pp(m) * <7(m)rOT2) *pyp',* de unde re *E2,* deci r e £j p|£2. Cum cp (r) = *p* rezultă că *p* G (Ex Q *E2).*

Invers, fie p G <p *(Er* Q *E2).* Există atunci r e 7} Q *E2* astfel ca *p* = = cp (r). Deoarece r G există m, *g(j)* G *N* şi *rjh* -G 7} cu (II.15.8) *r* = 6 (r01, *g* (0), r02) 6 (ru , *g* (1) *,_r12)...* 6 (rml, *g*(m), *rm2)p'.*

Dar, cum *r* G *E2* există n, *f(j)* G ÎV şi *rjh* s 7} cu

(11.15.9.) $r = x_0\ z_{/(0)}$ *ya* $(r_n\ ,\ f\ (1),\ r_{12})\ldots\ a\ (r_{nl}\ „\ f\ (n),\ r_{n2})$ *pyp'.*

Se observă căm $= n$ prin calcularea numărului de apariţiiale lui y în membrii drepţi aiegalităţilor (II.15.8.—9.). Apoi avem g (j)= $f(j),\ j = 0,\ m$

şi £01 ~ [r02 = e' J'eto) ▪ 3-0 '_'jl Pg(j) = [r]il ' '}2 = '}2 ' [z]U-1) 1 (lg(j-l) [z]G-1) 2 ▪ = [r]il [r]i2 = 'jl Pg(i) [r]j2J' — şî p - l'ml IqM 'm2 •

Obţinem astfel următoarea deducţie în gramatica G :

*

[T]0 ▪ [r]oi Pg (0) [r]02 [T]01 ?p(o) [r]02 ▪ [r]n $Pg(l)$ [r]12 [r](m-l) 1

7p(m —1) f(m—1)2 $Pg(m)\ \hat{}12\ \hat{}$ml $Qg(m)\ \hat{}m2\ P\hat{}$ prin Urmare $p\ ^e\ E,\ |$

BIBLIOGRAFIE[1']

ÂBRAHÂM, S.

[1] Some questions of plirase struclure grammars. Computaţional linguistics 4 (1965), 61—70 ABRAMSON, H.D.

[1] A note on left — recursive rulcs and partitioning of a recognition inatrix for syntax-di- rected translation. Nordisk. Tidskr. Informations beliandling (BIT) (1970), 1 — 5. AHO, A.

[1] Indexed grammars — An exlension of context — free grammars. J. Assoc. Compui. Macii. 15 (1968), 617-671. ALTMAN, E„ B.; BANERJI, R„ B.

[1| Some problems of finite representability. Information Control 8 (1965), 251—263. AMAR, V.; PUTZOLU, G.

f 11 ceneralization of regular events. Information Control 8 (1965), 56 — 63.

AHIIGMMOB, A. B.

[II O rpynnoBbix H3MKax. KnOepueTHKa, KiteB, 1971, K» 4, CTp. 18—24.

[2] BecnepeOopHMft anajin3 HBEIKOB **nporpaMirponaHHn MerojiM pa3BepTKM.**

Proceedings of Symopsium and Summer School of Mathematical Foundations of Computer Science, High Tatras 1973, p. 175 — 180. ARBIB, M., A.

[1] The algebraic thcory of inachines. languages and semigroups. Academic Press, New-York — London, 1968. BALTUSH, F.

[1] Maşini matematice şi gramatici generative. Stud. Cerc. Mat. 21. (1969), 831—876. BAR-HILLEL Y.

[1] Language and information. Addison —Wesley Publ. Comp., Reading, Mass., 1961. BAR, D.

[1] Zur Theorie der synlaktisclien Analysealgoritlimcn fiir kontextfeie Grammatiken I. Elektron. Informationsverarbeit Kybernetik 5 (1969), 171—205.

[2] Zur Theorie der syntaktischen Analysealgorithmen fiir kontextfrcic Grammatiken II. Elektron. Informationsverarbeit Kybernetik 5 (1969), 217 — 238. BARBAULT, M. C. ; DESCLES, J. P.

[1] ' Vers une formalisation des grammaires transformationnelles. Math. Sci.

Humaines No 34 (1971), 27-41, No 35 (1971).
BARTHOLOMES, F. ; HOTZ, G.
[1] Homomorphismen und Reduktionen linearer Sprachcn. Springer — Verlag, Berlin — New York, 1970. BEJIELJKHII, M. II.

[1] BecKOHTeKCTHbie H **«OMHitauiioHiibie rpaMMaTHKH** H **CBHaamiEie c** HHMM **aJiropmjiMUHec- Kne npofijieMM. KuoepHeTMKa KneB, 1967, CN? 4, CTp. 90—97.**

■) Se va consulta şi bibliografia din : CREANGĂ, I. ; REISCHER, C. ; SIMOVICI, D. : Introducere algebrică în informatică. * Teoria automatelor. Editura Junimea, Iaşi, 1973.
[2] MaTeMaTHHecKafl Mo^ejib npejţjio»<eHnH. Kn6epHeTMKa KiteB, 1971, JN» 2, CTp. 125—132.

[3] 06 ORHOM **cnoco6e onncamiH ynpaBJieHRH** B **KaTeropMHJibHux rpaMMHTHKax. Kn6epHe- THKa KneB 1971, JN» 4, CTp. 146—147.**
BELL, J. L. ; SLOMSON, A. B.
[1] Models and Ultraproducts. North Holland Publ. Amsterdam, 1969.
BIRKHOFF, G.
[1] Lattice theory. Amer. Math. Soc. Colloquium Publications Providence, Rhode Island, 1968.
BLUM, M.
[1] A machine — independent theory of recursive functions. J. ACM 14 (1967), 322 — 336. BOASSON, L.
[1] Cdnes rationnels et familles agreables de langages. — Application au langagc ă comp- teur. Thesis, Univ. de Paris, 1971. BOASSON, L. NIVAC, M.
[1] Transductions et familles de langages. Math. Sci. Humaines No 35, (1971), 31—37.
BOflHAPHyK, B. r.

[1] ABTOMaTM M CO6MTHH. **y«p. MaTeM. TKypnaji 14 (1962) 351—361.**

[2] CncTeMbi ypaBHeHUii B **ajire6pe coGbrrnii. ÎK.** BMHMCJI. **MaTeM. n MaTeM. <j)H3HKa 3 (1963) 1077—1088.**
[3] MeTpnnecKoe npocTpaHCTBe coObiTiiit. Kn6epHeTHKa (KneB) 1965, JN» 1, 24 — 27, JN» 4, 22—30.
BOOK, R.
[1] Time-bounded grammars and their language.
J. Comput. System. Sci. 5 (1971) 397 — 429.
[2] Terminal context in context-sensitive grammars. Cenler for Res. in Comput. Technol. Harvard Univ. Preprint no. 10 (1971).
BOOK, R. ; GINSBURG, S.
[1] Multi-steck-counter languages. Math. Systems Theory 6 (1972), 37 — 48.
BOOK, R. V. ; GREIBACH, S. A.
Quasi-realtime languages. Math. Systems Theorv 4 (1970), 97 — 111. BOOK, R. V. ; GREIBACH, S. A. : WAGBREIT, B.
[1] Time — and tape — bounded Turing acceptors and Al-'Ls.
J. Comput. System. Sci. 4 (1970). 606 — 621.
BOOK, R. ; EVEN, S. ; GREIBACH, S. ; OTT, G.
[1] Ambiguity in graphs and expressions.
IEEE Trans. Comput. C — 20 (1971) 144 — 153.
BOOK, R„ V. ; GREIBACH, S„ A. ; IBARRA, O. H. ; WEGBREIT, B.
[1] Tape-bounded Turing acceptors and principal AFLs., J. Comput. System. Sci. 4 (1970), 622-625.
BOURBAKI, N.
(1) Theorie des ensembles : Elements de mathematique. Herinann, Paris, 1970.
BOUCHER, C.
[1] Fermeture de certaines classes de langages formels sous des parmutations

lingvistiques. Rev. Franţaise Informat. Recherche Operationnelle 3 (1969), Ser. R. — 2, 51—61. BRAINERD, B.
[Ij An analog of a theorem about context free languages. Information Control 11 (1968), 561—567.
[2] Introduction to the mathematics of language study. American Elsevier Publ. Co., New York, 1971.
BURKHARD, W. A. ; VARAIYA, P. P.
[1] Complexity problems in real time languages. Information Sci. 3 (1971), 87 — 100.
CANTOR, D„ C.
[1] On the ambiguity problem of Backus systems. J. ACM 9 (1962), 477 — 479.
CARACCIOLO, A.
[1] Some remarks on the syntax of symbolic programming languages. Corn. ACM 6 (1963), 456-460.
CĂRĂUŞU, A.
[1] On a natural ordering of rules in context-sensitive grammars. Preprint of the XI[4]* Internaţional Congress of Lingvists. Bologna — Firenze, 1972, 1229 — 1236.
[2] On ordered rewriting systems. Cahiers de ling, thdor. et appl. 9 (1972), 277 — 284. CARLUCCI, L.
[1] A formal system for texture languages. Pattern Recognation 4 (1972), no. 1, 53 — 72. CHOMSKY, N.
[1] A note on phrase structure grammars. Inforination Control 2 (1959), 393 — 395. CÎŞLARU, C.
[1] Configuraţii de ordin infinit. Stud. Cerc. Mat. 21 (1969), 561—570. COCHET, Y. ; NIVAT, M.
[1] Une gincralisation des ensembles de Dyck. Israel J. Math. 9 (1971), 389-395. COLMERAUER, A.
(1) Total precedence relations. J. Assoc. Comput. Mach. 17 (1970), 14—30.
CONWAY, J. M.
[1] Regular algebra and finite machines. Chapman and Hali, London, 1971.
CRĂCIUN, C. V.
[1] A-domination and conditioned domination in algebraic grammar. Stud. Cerc. Mat. 20 (1968), 683-695.
[2] Unele proprietăţi ale operatorilor contextuali din lingvistica algebrică. Stud. Cerc. Mat. 22 (1970), 419-434.
КРІОКОВ, IO. A.
[1] MauiHHM TiopHHra c TpeMH СОСТОНННННМН М «ByMH CHMOoJiaMn n О«ННМ cocTOHHne.M n Н CHMfiojiaMii. KnBepHeTMKa KueB 1971, N» 1, CTp. 12—13.
CREANGĂ, I. ; REISCHER, C. ; SIMOVICI, D.
[1] Introducere algebrică In informatică * Teoria automatelor. Editura Junimea, Iaşi, 1973.
CUDIA, D. ; SINGLETARY, W.
[11 Degrees of unsolvability in formal grammars. J. Assoc. Comput. Mach. 15 (1968) 680, 692.
QULIK, K. II.
[1] n-ary grammars and the description of mapping of langauges. Kybernetika (Prague) 6 (1970), 99 — 117.
DAVIS, M.
[1] Computability and unsolvability. Mc Graw-Hill, New York, 1963. DINCĂ, AL.
[1) Ecuaţii diofantice şi limbaje context-free. Studii Cerc. Mat. 22 (1970). 1059 — 1172.
[2| Distanţe contextuale In lingvistica algebrică. Studii Cerc. Mat. 25 (1973), 223 — 265.
[3] Distanţe şi diametre Intr-un semigrup (cu aplicaţii la teoria limbaielor). Studii Cerc. Mat. 25 (1973), 359-378.

DUYVESTEYN, A. J. W.
[1] Sentence analysis. Math. Centrum. Amsterdam Afd. Zuivere Wisk. 1969, ZW— 005. EICKEL, J. fLOECKX, J.
[1] The relation between derivations and syntactical structures in phrase-structure grammars. J. Comput. System Sci. 6 (1972), 267-282.
EL GOT, C„ C.
[1] Review of "A remark on finite transducers", IRE Trans. Electron, comput. EC 11 (1962), 802.
ENESCU,GH.
[1] Logică simbolică, Ed. ştiinţifică, Buc. 1971. ERSHOV, A., P.
[1] Programming Programme for the BESM Computer. Pergamon Press, New York, 1959. FABIAN, V.
[1] Structural unambiguity of formal languages. Czech. Math. J. 14 (1964), 394 — 430.
FISCHER, M-
[1] Two characterizalions of context sensitive languages. IEEE Conf. Record 10 tli Ann. Symp. Switching Automata Theory (1969), 1 19 — 156).
FLIESS, M.
[1] Transductions algebriques. Rev. Franţaise Informat. Recherche Operalionelle 4 (1970) S6r. R - 1, 109-125.
FLOYD, R.W.
]1] On the nonexislence of a plirase structure grammar for ALGOL (10. GACM. 5 (1962), 483-484.
[2] On atnbiguity in plirase structure languages. GACM 5 (1962), 526 — 534.
[3] Syntactic analysis and operator precedence. .1. ACM 10, (1963), 316 — 333.
14] Bounded context syntactic analysis. GACM 7, (1964), 62 — 67.
|5) New proofs and old theorems in logic and formal Jingvistics. Computer Associates Inc., Wakefickl, Mass, 1961.
[6] The syntax of programmir.g languages — a survcy. PGEC, 13 (1961) 316-353.
FRIANT, J.
[1] Grammaires ordonnies — grammaires matriciellcs. Univ. de Monlreal MA — 101 (1968).
[2] I.angages ultralinâaires et superlindaires. Nouvelles caractei isations. Univ. de Monlreal MA - 102 (1968).
FRIS, I.
[1] On stop — conditions in the definitions of constructive languages. Zeitscher. f. Math. I.ogik und Grundlagen d. Math. 11 (1965), 61—73.
[2] Grammars with parţial ordering of the rules. Information Control 12 (1968), 415—125. FUJII, M. ; KASANI, T.
[1] Some structural equivalence relations and transformation of context-free grammars. Electron C.ommun. Japan 52 (1969), 162 — 170.
I'UJITA, T.
[1] A note on recursive language. Information Precessing in Japan 8 (1968), 12 — 13. GABRELIAN, A.
[1] Pure grammars and pure languages. Univ. of Waterloo, Dcp. of AppL Anal, and Compui. Sci. Res. Rep. CSRR 2027 (1970).
GAIFMAN, H.
[1] Dependency systems and plirase structure systems. Information Control 8 (1965), 304 —337 G1LL, A.
[1] Introduction to the theory of finite-state machines. Mc Graw Hill, New York, 1962. GINSBURG, S.
[1] An introduction to mathematical machine theory. Addison — Wesley, Reading, Maşs. 1962.
[2] The mathematical theory of context-frce languages. Mc Graw-Hill Book Co„ New York,1966. GINSBURG, S. , GREIBACH, S.
[1] Abstract families of languages. Studies in Abstract Families of Languages. Mem. Amer. Math. Soc. N° 87, 1—32. Amer. Math. Soc. Providence, R. I.

1969.
[2] Principal AFL. J. Comput. System. Sci. 4 (1970), 308 — 338. GINSBURG, S. ; HARR1S0N, M. A.
[1] Bracketedcontext free languages. System Devei. Comp. Rept. TM— 738/023/00. Jan. 4,1966. [2j Bracketed context-free languages. J. Comput. System Sci. 1 (1967), 1—23.
GINSBURG, S. ; HOPCROFT, J.
[1] Images of AFL under ccrtain families of homomorphisms. Math. Systems Theory 5 (1971), 216-227.
GINSBURG, S. ; PARTEE, B.
[1] A mathematical model of transformationel grammars. Information Control 15 (1969), 297-334. •
GINSBURG, S. ; ROSE, G.F.
[1] On the existencc of gencralors for certain AFL. Information Sci. 2 (1970), 431 — 116. GINSBURG, S. : SPAN1ER, E.H.
[1] Mappings of languages by twotape devices. J. ACM, 12 (1965), 423 — 434.
[2] Control sets on grammars. Math. Systems Th. 2 (1968), 159 — 177.
[3] Derivation-bounded languages. J. Comput. Sci. 2 (1968), 228 — 250.
[4] Substitution in families of languages. Information Sci. 2 (1970), 83 — 110.
[5] AFL with the semilinear property. J. Comput. System. Sci. 5 (1971), 365 — 396. GINSBURG, S.; GREIBACH, S. ; HOPCROFT, J.
[1] Pre — AFL. Studies in Abstract Families of Languages. Mem. Amer. Math. Soc.. N» 87, 41 — 51. Amer. Math. Soc. Providence, R.I., 1969.
GINSBURG, S. ; HIBBARD, T. N.; ULLIAN, J. S.
[1] Sequences in context free languages. III. J. Math.. 9 (1965), 321 —337. GINZBURG, A.
[1] Algebraic theory of automata. Academic Press, New York, 1968. rJIAflKHfl, A. B. i
[1] flbop.Ma.JLHMe rpaMMaTMKn n HSMKII, MocKna, 1973. rJIA^KHH, A. B. ; MEJIbHYK, H. A.
[1] BjieMeHTtr MaTeMaTHRecKoii jînrBncTiiKM. Hayna, Mocima, 1969. rJIEBCKWll, IO. B., KOrAH, fl. M.
[1] AJWHTHBHO ynpaBJiHeMue CMCTCMEI n H3HKH : HCKOTopbie a.iropn<J>HMMHecKHe npo6- neMbi. KnCepHeTHKa KiieB 1971, AC» 4, CTp. 25—38.
GREIBACH, S. A.
[1] Formal parsing systems. C. AC M 7 (1964), 499 — 504.
[2] A note on code sets and context frec languages. Math. Lingvistics an<) Automatic Translation, Harvard Univ. Computation Lab. Rept. NSF 15, 1965, II —1—II —5.
[3] A new normal forrn theorem for context-free phrase structure grammars. J. ACM, 12 (1965), 42-52.
[4] A note on undecidable properties of formal languages. Math. Systems Theory 2 (1968), 1 — 6.
[5] An infinite hierarchy of context-free languages. J. Assoc. Comput. Mach. 16 (1969), 91 — 106.
[6] Full AFL's and nested iterated substitution. IEEE conf. Record lOth Ann. Symp, Switching Automata Theory (1969), 222—230.
[7] Characteristic and ultrarealtime languages. Information Control 18 (1971), 65 — 98. GREIBACH, S. ; GINSBURG, S.
II] Multitape AFA. J. Assoc. Comput. Mach. 19 (1972), 193 — 221. GREIBÂCH, S. , HOPCROFT, J.
(1] Scattered context grammars. J. Comput. System Sci. 3 (1969), 233—247.
[2] Indcpendence of AFL operations. Studies in Abstract Families of Languages. Mem. Amer. Math. Soc. No. 87. 35 — 40. Amer. Math. Soc. Providence, R.I., 1969.

GRIFFITHS, T. V. ; PETRICK, S. R.
[1] On the relative efficiencies of contex-free grammar recognizers. C. ACM, 8 (1965), 289-300.
GROSS, M. ; LENTIN, A.
[1] Introduction to formal grammars. A Springer-Verlag, New York Berlin, 1970.
GRUSKA, J.
[1] On structural unambiguity of formal languages. Czech. Math. J. 15 (1965), 283 — 293.
[2] Introduction in formal languages.- Some properties of reducing transformations and of ' isolable sets. Czech. Math. j. 15 (1965), 406 — 414.
[3] A characterization of context free languages. J. Comput. System Sci. 5 (1971), 353 — 364.
[4] A few remarks on the index of context - free grammars and languages. Information Control 19 (1971), 216-223.
GRZEGORCZYK, A.
[1] Some classes of recursive functions. Rozprawy Mat. 4 (1953), 1 — 45.
HARRISON, M„ A. ; SCHKOLNICK, M.
[1] A grammatical characterization of one ■ way nondeterministic stack languages. J. Assoc. Comput. Mach. 18 (1971), 148-172.
HAVEL, I.
[1] On one — side context — sensitive grammars. In J. Diirr and G. Hotz (eds). Automaten theorie und formale Sprachen. Bibliographisches Inst. Mannheim, 1970,221—225.
[2] Strict deterministic languages. Univ. of California (Berkeley), Comput. Sci. Dept. Tecii. Rep. No. 1, 1971.
HERMAN, G. T.
[1] Linear regular languages. I. Acta Cybernet. 1 (1969), 3 — 12.
[2| The computing ability of a developmental model for filamentous organisms. J. Tlleoret. Biol. 25 (1969), 421-424.
[3] Closure propcrties of some families of languages associated witli biologica! syslems. State l'niv. of New York at Buffalo, Dept. of Comput. Sci. Rep. 7 — 71. 1971.
HOTZ, G. ; SCHNORR, C.P. (eds)
[1] Tagung iiber Formale Sprachen. Gesellschaft fur Mathematik und Ilatenverarbeitung Bonn, 1970.
IBARRA, O. H.
[1) Simple matrix languages. Information Control 17 (1970), 359 — 391. IRONS, E.T.
11] A syntax directed compiler for ALGOL 60 CACM, *i* (1961), 51 — 55. JĂRVI, T.
|1] On control sets induced by grammars. Ann. Acad. Sci. Eenn. Ser. A, I, (1970), 7. JONES, N. D.
[1| Notes on the theory of Computability. Pennsylvania State Unis, 1968.
[2| Context-free languages and rudimentary attributes. Math. System Theory 3 (1969), 102 — 109.
[3] A note on the index of a context-free language. Information Control 16 (1970), 201 —202.
JURCA, M.
[11 Asupra relaţiei dintre tipul si Q-clasa unui cuvînt. Studii. Cerc. Mat. 22 (1970), 1033 — 1040. KAMEDA, T. . VOLLMAR, R.
[1] Note on tape reversal coinplexity of languages. Information Control 17 (1970), 203 — 215. KAMÎNGER, F.
[1] The noncomputability of the channel capacity of context-sensitive languages. Information Control 17 (1970), 175-182.
KASAI, T.
[1] An hierarchy between context free and context-sensitive languages. J. Comput. System Sci. 4 (1970), 492-508.
KASAMI, T.; TANIGUCHI, K.

[1| Some decision problems of formal languages. Electron. Coininun. Japan. 51 (1968), 151 — 158. KLEENE, S.C.
[1] Introduction to inetamathematics. D. Van Nostrand, Princeton, New Jersey, 1952. KNUTH, D.E.
[1| Top-down syntax analysis. Acta Informatica 1 (1970), 79 — 110. KOBAYASHI, K.
[1| Some unsolvable problems concerning context-free languages and their application to on-line language recognizers. Electron. Commun. Japan 51 (1968), 93 — 97.
KOPRIVA, J.
[1) Decomposition translations and svntax translation schcinata. Kybernetika (Prague) 6 (1970), 403-417.
KOSTER, C. H. A.
[11 Two-level grammars. Seminar on Automata Theory and Mathcinatical Lingvistics. L'trecht, Abstract no. 11 (1970).
KRÁL, J.
[1] On multiple grammars. Kybernetika 1 (1969), 60 — 85.
[2] A modification of a substitution theorem and some necessary and sufficient conditions for sets to be context-free. Math. System. Theory 4 (1970), 129 — 139.
KUICH, W.
(1) Systems of pushdown acceptors and contcxt-free grammars. Elektron. Informationsverar- beit. Kybernetik 6 (1970), 95 — 113.
[2] A context-free language and enumeration problems on infinite trees and digraphs. J. Combinatorial Theory Ser. B. 10 (1971), 135 — 142.
KUICH, W. ; MAURER, H.
[1] On the inherent ambiguity of simple tuple languages. Computing (Arch. Elektron. Rechnen) 7 (1971), 194-203. KURKI-SUONIO, R.
[1] Notes on top-down languages. BIT 9 (1969), 225 — 238.
[2) A programmer's introduction lo coinputability and formal languages. Studentlitteratur, Sweden, 1971.
I. EARNER, A. :I.1M, A.L.
[1] A note on transforrning context-free grammars to Wirth-Weber precedence form. Comput.
J. 13 (1970), 142-111.
LEHMA NN, D.
[1] *L.H (k)* **grammars and deterministic languages. Israel, J. Math. 10 (1971), 526 — 530. JIETHMEBCKKfl, A. A.**
[1] IlpejtcTaBJieHHe KOHTeKCTHe-CBo6onHMX HBMKOB B **anTOMaTax c naMHTbio Tuna pushdown. KnOepHeTMKa, KneB 1, (1965) 80—84.**
[2] O nociienoBaTeJibHOM pacuinpeHHH a.'iropiiTMHHecKitx HBMKOB. **KnCepHeTiiKa, KtieB 1967, X» 6, CTp. 1—6.**
LEWIS, P. M. II ; STEARNS, R.
[1] Svntax directed transduction. J. Assoc. Comput. Mach. 15 (1968), 464 — 488.
LINDENMAYER, A.
[1] Mathematical models for cellular interactions in development, parts I—II. J. Theoret. Biol. 18, (1968), 280-315.
LOECKX, J.
[1] Mechanical construction of bounded-context parsers for Chomskv-o-tvpe languages. Philips Res. Rep. Suppl. Nr. 3 (1964) V + 92 p.
[2] Grammaires formalles et automates. Bull. Soc. Math. Belg. 21 (1969), 145 — 165.
[3] The parsing for general phrase — structure grammars. Information Control 16 (1970), 443-464.
LYN'CH, W. C.
[1] Ambiguities in Backus form languages. Doctoral dissertation, Univ. of Wisconsin, Madison Wisc. 1963.

MAGIDOR. M. . MORAN, G.
[1] Probabilistic tree automata and context free languages. J. Math. 8 (1970), 340 — 348. MAL'CEV, A.J.
[1] Algebraiceskie sistemy. Nauka, Moskva, 1970. MAJIblJEB, A. 14.
[1] AjiropitTMM n peKvpctiBHMe (JniHRiţnn. I<3H. «HayKa» $H3—MBT. **jiMTepaTypM. MocKBa, 1965.**
MALUS7.YNSKI, T.
[1] The lattice of subgrammars of context free grammars. Algorytmy 7 (1970), 21 —27.
MARCU, E.T.
[1] On an operator that is related to Sestier's operator in contextual algebraic analysis. Stud. Cerc. Mat. 21 (1969), 499-508.
MARC.US, S.
[1] Gramatici şi automate finite. Edit. Academiei R.S.R., Bucureşti, 1964,
[2] Contextual grammars. Rcv. Rumaine Math. Pures Appl. 14 (1969), 1525 — 1534.
MATTHEWS, G. H.
[1] Discontinuity and asymmetry in phrase structure grammars. Information Control 6 (1963), 137 — 146.
[2] A note on asymmetry in phrase structure grammars. Information Control, 7 (1964), 360 — 365.
[3] Two-way languages. Information Control 10 (1967), 111 — 119.
MAURER, H.
[1] Theoretische Grundlagen der Programmiersprachen. Hochschul taschenbiicher 404, Bibliographisches Inst., 1969.
[2] A direct proof of the inherent ambiguity of a simple context free languages. J. Assoc. Comput. Mach., 16 (1969), 256-260.
MAYER, O.
[1] Some restrictive devices for context-tree grammars. Information Control 20 (1972). 69 — 92. Mc NAUGHTON, R. ; PAPERT, S.
[1] Counter-free automata. Research Monograph no. 65. M.T.I. Press, Cambridge, Mas. 1971.
McVVHIRTER, I.
[1] Substitution expressions, J. Gomput. System Sci. 5 (1971), 629 — 637.
MEALY, Ci. H.
[1] A method for synthetizing sequential circuits. Beli. Syst. Tecii. J. 34 (1955) 1045 — 1079. MIHALYI, M.
[1] Despre noţiunea de productivitate sintactică. Stud. Cerc. Mat. 21 (1969), 419 — 431. MILGRAM, D. L. ; ROSENFELD, A.
[1] A note on scattered context grammars. Information Precessing J.ett. 1 (1971/1972), 17 — 50. ĂţlNSKY, M. ; PAPERT, S.
[1] Unrccognizable sets of numbers. J. Assoc. Gomput. Mach. 13 (1966), 281 - 286. M1ZUM0T0, M. ; TOYODA, J. ; TANAKA, K.
[1] Some considerations on fuzzy automata. Electron. Commun. .Japan 52 (1969), 93 — 99. MORSE, M. : HEDLUNG, G.
[1] Symbolic dynamics. Ainer. J. Math. 60 (1938), 815 — 866.
NASH, B. ; COHEN, R. ,
ţl| Parallel leveled grammars. IEEE Gonf. Record 10 tji Ann. Symp. Switcliing Automata Theory (1969), 263-276.
NASU, M. ; HONDA, N.
[1] A contcxt-free language whicli is not acceptable by a probabilistic automaton. Informa- tion Control 18 (1971), 233-236.
NAVRATIL, E.
[1] Context-free grammars with regular conditions. Kybernetika (Prague) 6 (1970), 118 — 126. NEBERSKY, L.
[1] On one of the boundaries of the Constituent conception of grammar, Kybernelika (Prague) 7 (1971), 227-229.
NISHIZAWA, T.

[1] An algebraic proof of Kleenc-Myhill's equivalence theorem. Mein. Fac. Sci. Kypshu Univ. Ser. A, 24 (1970), 94-99.
[2] A note on an ordering of nonrepeated word sequences. Meni. Fac. Sci. Kyushu Univ. Ser. A, 24 (1970), 104-109.
[3] Sequential machines with general counters. Mein. Fac. Sci. Kyushu Univ. Ser. A, 24 (1970), 110-119.
N1VAT, M.
[1] Transduction des langages de Chomsky. Ann. Inst. Fourier, Grcnoble 18 (1968), 339 — 455. NOVOTNY, M.
[1] Algebraic slructures of mathematical lingvistics. Bull. Math. Soc. Sci. Math. R. S. Rou-
mâine (N.S.) 12 {60} (1968), 87-101.
[2] On configurations in the sense of Kulagina.
Spisy Prirod. Fak. Univ. Brno, 1969, 301—315.
[3] On two classes of languages. Spisy Prirod. Fak. Univ. Brno, 1970. Cislo 518, 465 — 470 (1971).
[4] Complete characterization of classes of Chomsky by means of configurations. Acta F.R.N.
Univ. Comen. — Mathematicâ — Mimoriadne, cislo (1971), 63 — 71.
[5] On some relations defined by languages.Prague Studies in Math.Linguistics 4 (1970),157 —170.
[6] On homomorphism of norm bounded languages. Prague Studies in Math. Linguistics 5 (1973).
[7] On some operators reducing generalized grammars. Matematicky ustav CSAV, pobofka Brno, 1973.
OGDEN, W.
[1] A helpful result for proving inherent ambiguity. Math. Systeme Theory 2 (1968), 191 — 191. ORMAN, G.
[1] O condiţie de existentă a mulţimilor omologe. Bul. Inst. Politehn. Braşov, Ser. A. Mec. 11 (1969), 24-27.
[2] Considerations sur Ies langages „context-free" lin6airs â gauche et a droite. Rev. Rou- maine Math. Pures Appl. 16 (1971), 1107-1114.
PAIR, C.
[1] Sur Ies notions algdbriques li6es â l'analyse syntaxique. Rev. Franţaise Informat. Ope- rationnelle 4 (1970) Ser. R-3, 3—29.
PAULL, M. ; UNGER, S.
[1] Structural equivalencc of context frec grammars. *J.* Comput. Svstem Sci, 2 (1968), 427 — 46.3. PAZ, A.
fl] Formal series, finitcness properties and decision problems. Ann. Acad. Sci. Fenn. Ser. A, I No -19.3 (1971), 16 p.
PETER, R.
[1| Bemerkungen zu den G. R6v6rzschcn terminalen Grammatiken. Ann. C'niv. Sci. Bu- dapest. Eiitros Sect. Math. 11 (1968), 109-113.
PETERS, P. S. Jr. : RITCHIE, R.
[1] On restricting the base component of transformational gramniars. Informalion Control 111 (1971), -18.3-501.
' PETRO NE, L.
[1] On prccedencc gramniars. Olivetti Electron. Res. I.ab. Rept. Febr. 1965.
POPESCU, E.
[1] Asupra unui model algebric al părţii de vorbire. Studii şi Cerc. Mat. 23 (1971), 1151—1160. POST, E. I..
[1] Formal reduclion oT the general combinatoria! decision problem. Amer. J. Matli. 65 (1943), 197-215.
RAISCHI, C. T.
il] Relaţia de dominare şi analiza algebrică contextuală. Stud. Cerc. Mat. 20 (1968), 31—41.]2] Contexte caracteristice în lingvistica algebrică.
Stadii şi cercetări Mat. 21 (1969), 123 — 139.
[3] Operatorii lui Ncbesky si contextele caracteristice. Stud. Cerc. Mat. 22

(1970), 891-916.
fl] Absolutely parallel gramniars and two-way finite-state transducers. J. Coinput. System Sci. 6 (1972), 321-342.
PEflbKO, B. H.
[1] K cnHTaKciiuecKOMy aHajmay KoiiTeKCTiie CBoOogmjx НЗNKOB. Kn6epneTHna KneB 1966, A» 3, CTp. 52—63.
REISCHER, C. ; SIMOVICI, D.
[1] Limbaje litere de context. Universitatea „Al. I. Cuza“ — Iaşi, Centrul de multiplicare, 1973.
RfiVfiSZ, GY.
[1] Syntactic analysis and unilateral conlcxt-scnsitive grammars. Studia Sci. Matli. Hungar,
4 (1969), 267-278.
[2] L’nilateral context sensitive grammars and left-to-right parsing. J; Comput. System. Sci.
5 (1971), 337-352.-
[3] Dual pushdown automata and context sensitive grammars. Acta Cybernet. 1 (1972), 147-157.
ROGERS, H.
[1] Theory of recursive functions and effective computabilîty. McGraw Hill Book Co. New York, 1967.
ROSENBERG, A. I..
[1] A machine realizalion of the linear context-frec languages. Informalion Control 10 (1967), 175-188.
[2] A note on ambiguity of context-free languages and presentations of semilinear sets. J.A.C..M. 17 (1970), 44—50.
ROSENFELD, A.
[1] Isotonic grammars, parallel grammars, and picture grammars, Machine lntelligence, 6,
281 — 294, American Elsevier, New York, 1971.
ROSENKRANTZ, D.
[1] Programmed grammars and classes formal languages. J. Assoc. Comput. Mach. 16, (1969), 107-131.
ROSENKRANTZ, D.J.; STEARNS, R.E.
[1] Properties of deterministic top-down grammars. Information Control 17 (1970), 226 — 256.
ROZENBERG, G.
[1] On the introduction of orderings into the grammars of Chomsky’s hierarchy. Bull. Acad. Polon. Sci.Sir. Sci. Math. Astronom. Phys. 17 (1969), 559—563.
[2] TO L — systems and languages. EROU Piiblicaties (Utrecht) no, 103, 1971.
[3] On OL-systems with restricted use of productions. ERCU Publicatics (Utrecht) no. 116, 1971.
[4] I.-systems with interactions ERCU Publicaties (Utrecht) no. 120, 1971.
[5] Directioncontrolledprogrammedgrammars.Acta Informatica 1 (1972), 242 — 252. ROZENBERG, G.; DOUCET, P.
[1] On OL-languages. Information Control 19 (1971), 302 — 318. SALOMAA. A.
[1] Onfiniteautomata with a time-variant structure. Information Control 13 (1968), 85 — 98.
[2] On grammars with restricted, use of productions. Ann. Acad. Sci. Fennicae. Ser. AI 454 (1969).
[3] On the index of a context-free grammar and language. Information Control 14 (1969), 474 — 477.
[4] Periodically time-variant context free grammars. Information Control 17 (1970), 294 — 311.
[5] Matrix grammars with a liftmost restriction. Information Control 20 (1972), 143—149.

[6] Formallanguages. Academic Press. New York, Londen, 1973.
SALWICKI, A.
[1] Formalised algorithmic languages. Bull. Acad. Polon. Sci. Sdr. Sci. Math. Astronom. Plivs. 18 (1970), 227-232.
[2] On the equivalence of FSexpressions and programs.
Bull.Acad.Polon. Sci.Sfr. Sci.Math. Astronom. Phys. 18 (1970), 275 — 278.
CAMOftJIEHKO, JI. T.
[1] O cBOilcTBax rpaMMaTMK HenocpeflCTBeHHhix cocTaBJimomnx.
Kn6epHeTHKa KneB, 1966, 5, CTp. 24—27.
[2] O MeTOJte nOCTpOeHMH M CBOftCTBaX K0HTeKCTH03aBHCHMMX rpaMMaTMK H H3HK0B . Prceedings of Symposium and Summer School of Mathematical Foundations of Computer Science, High Tatras 1973, p. 307 — 312.
SCHORRE, D. V.
[1] A necessary and sufficient condition for a context-free grammar to be unambiguous. System Development Corp. Rept. SP-2153, 1965,
SCHUTZEMBERGER, M. P.
|1] Some remarks in Chomsky's context-free languages. M. I. T.
Res.Lab.Electron.Quart.Prag.Rept.,1961.
[2] On a theorem of R. Jungen. Proc. Amer. Math. Soc.13 (1962), 885—890.
[3] Classification of Chomsky'slanguages.Proc. IFIP. WorkingConference on Formal Language Description Languages. Baden, Austria, 1964.
SHAMIR, E.
[1] A remark on discovery algorithms for grammars. Information Control 5 (1962), 246—251.
[2] Mathematical Models of Languages. Proceedings of IFIP Congress 65,
Spartan Borks, Inc. Washington, D. C. 1965, 71—75.
[3] Some inherently ambiguous context free languages. Information Control 18 (1971), 355 — 363.
SIMOVICI, D. A.
|1] On some measures on free semigroups induced by semiautomate. Proceedings of Symposium and Summer of Mathematical Foundations of Computer Science, High Tatras 1973, p. 313-316.
SIROMONEY R.: SIROMONEY, G.
[1] Equal matrix languages and Kleene closure.
Proc. Indian Acad. Sci. Sect. A 74 (1971), 172 — 188.
SKOWRON, A.
[1] A syntactic translation of the machinedetermined languages. Bull. Acad. Polon. Sci. Math. Astronom. Phys. 18 (1970), 603 — 607.
|2] Semantic translation of programming languages. Z. Math. Logik Grundlagen Math. 17 (1971), 39-46.
f 3] Languages detenninedby machinessystems. Bull .Acad. Polon. Sci. Ser. Sci. Math. Astronom. Phys. 19(1971), 321-326.
SMITH, AV. B.
[1] Error detection in formal languages. J. Compu t. System Sci. 4 (1970), 385 —105. SMULLYAN, R.M.
[1] Theory of formal systems. Princcton Univ.Press.Princeton, New Jersey, 1961.
SOLOMON', S. Y.
[1] Lingvistica algebrică şi teoria modelelor. Stud.Cerc.Mat. 21 (1969), 1007 — 1134. SZOCS, H.
[1] O metodă de transformare a proceselor structural omogene in mulţimi statistice. Stud. Cerc. Mat. 20 (1968), 259 - 265.
IIIPElîTJEP, IO. A.
[1] O BapnaijnoHHEix npMHpnnax B jmHrBMCTMKe. KiiGepHeTMKa KneB, 1966, JVÎ **2, CTp. 49—56.**
TANAKA, E.; KASAJ, T.: TARI, I.

[1] Self-learning of finite state language by machines. Electron. Commun. Japan 52 (1969), 115-122.
TANIGUCHI. Ii.; KASAMI, T.
[1] Reduction of context-free grammars. Information Control 17 (1970), 92 — 108.- THATCHER, J. XV.
[1] Characterizing derivation trees of context-free grammars through a generalizat ion of finite au tomata theory. J. Compu t., Sy s tem Sc i. 1(1967), 317 — 322.
TH1ERRIN, G.
[1] Decomposition des langages rcguliers. Rev. Franţaise Informat. Recherchc Operalicnnelle
3 (1969), Ser. R-3, 45-50.
THUE, A.
[1] Probleme iiber Verănderungen von Zeichcnreihen nach gegcbencn Regcln. Skriftei utgit av Videnskapsselskapet i Kristiana 1 10 (1914), 34 p.
ULLIAN, J.S.
[1] I-'ailure of a conjecture about context free languages. Information Control 9 (1966), 61—65. L'LLMAN, J. D.
[1] Pushdownautomata wilh boundcd backtrack.System Devol.Corp. Rept.TM — 738/022/00 Dec. 1965.
UREONEN, T.
[1] On axiom systems for regular expressions and on equations involving languages. Ann. Univ.Turku. Ser. A I No 145 (1971), 52 pp.
VAIDA, D.
Jl] Algoritmi de compilare — cu aplicaţii la limbaje de tip ALGOL. Editura didactică şi pedagogică, Bucureşti, 1971.
VERE, S.
[1] Translation equations. Comm. ACM 13 (1970), 83 —89. WALTER, H
[1] Verallgemeinerte Pullbackkonstruktion bei Semi-Thuesystemen und Grammatiken. Elek- tron. Informationsverarbeit. Kybernetic 6 (1970), 239 — 254.
WALTERS, D.
[1] Deterministic context -sensitive languages. I and II. Information Control 17 (1970), 14 — 61. WEGBREIT, B.
•1] A generator of context-sensitive languages. J. Comput. System. Sci. 3 (1969), 456 — 461. XVOOD, D.
[1] A note on top-down deterministic languages. Nordisk. Tidskr. Informationstebehaudling (BIT) 9 (1969), 387 — 399.
[2] The normal form theorem — another proof. Comput. 12 (1969), 139 — 147.
[3] A gcneralizcd normal forin thcorcm for conlcxl-frcc grammars. Compui. J. 13 (1970), 272-277.
YNTEMA, M.
11] Inelusion relations among familics of context-frec languages. lnformation Control 10 (1967), 572 — 597.
|2| Cap expressions for context-frec languages. lnformation Control 18, (1971), 311 — 318. ZADED, L.A.
[1] Quantitative fuzzy semantics. lnformation Sci. 3 (1971), 159 — 176.
ZEI.INKA, B.
[1] Sur le PS-isomorphisme fort des langages.
BulI.Math. Soc. Sci. Math. R.S. Roumanie (N.S.) 13 (61) (1969), 123-126.
[2] Les graphes de Hasse des moddles analitiques de langages.
BulI.Math. Soc. Sci. Math. R.S. Roumanie (N.S.) 13 61 (1969), 389-392.
[3] Le groupe des S-automorphismes d'un langage.
Rev. Roumaine Math. Pures Appl. 15 (1970), 453 — 458.
[4] Deux probldmes sur les grammaires â un nombre d'ctats fini.
Rev. Roumaine Math. Pures Appl. 16 (1971), 999—1002.
ZIMMER, R.

[1] Soft precedence. lnformation ProcessingLett. 1 (1971), 108—110.

INDICE ALFABETIC

Alfabet 28
alfabet de ieşire 73
alfabet de intrare 73
alfabet neterminal 65
alfabet terminal 65
algebră Boole 26
algebră Boole B_2 26
algebră Lindenbaum 58
aplicaţie 11
aplicaţie bijectivă 11
aplicaţie contactantă 32
aplicaţie de marcare 18
aplicaţie injectivă 11
aplicaţie surjectivă 11
arbore 17
arbore de deducţie 86
arc în graf 16
arc în hipergraf 18
aritatea unei operaţii 11
automat determinist 73
automat de tip Leticevski 123
automat finit 73
automat iniţial 73
automat liniar mărginit 141
automat nedeterminist 73 •
automat neiniţial 73
automat pushdown 113
automat slab iniţial 73
axiomele calculului propoziţional 55

Bijecţie 11 buclă 17
Calcul al unei configuraţii 151
cap de citire-scriere 113, 150
cel mai mare element, 13 cel mai mic element 13 clasă de echivalenţă 13 coeficient al unui term 17 componentă conexă 17 componentă conexă a unui hipergraf 19 complementara unei mulţimi 11 compunerea de maşini Turing 165 concatenare 28 conectiv logic 54 configuraţie instantanee a unui *ALM* 142
configuraţie instantanee a unui *APD* 114
configuraţie instantanee a unei *MT* 151
configuraţie instantanee a unei *MTBI* 154
configuraţie instantanee ' a unei *MTMMB* 158
configuraţie instantanee precede o configuraţie instantanee în- tr-un *ALM* 142
configuraţie instantanee precede o configuraţie instantanee în- tr-un *APD 114*
configuraţie instantanee precede o configuraţie instantanee într-o *MT* 151
configuraţie instantanee terminală 164
congruenţă 28
congruenţă la dreapta 28

congruenţă la stînga 28
corespondenţa biunivocă 11
corpul *GF* (2) 26 cuantificator
existenţial 58 cuvînt 28
cuvînt generat direct într-o
gramatică 65
cuvînt generat într-o
gramatică 66 cuvîntul nul 28
cuvînt recunoscut de un
automat prin un semnal de
ieşire 75

Deducţie 66
deducţie imediată la stînga 90
deducţie imediată la dreapta
90 derivare la stînga în raport
cu *i* 124
descendent al unui vîrf 17
descendent direct al unui vîrf
17 descendent imediat 17
descendent impropriu 17
descendent în hiperarbore 19
descendent propriu 17
diferenţa de mulţimi 11
diferenţa simetrică de
mulţimi 11 dispozitiv de
comandă 113, 150 drum de
lungime n într-un hipergraf
19 drum într-un graf 16

Echivalenţa nucleară a unei
aplicaţii 16
ecuaţie universală 37 element
complementar 25 extremitate
finală a unui arc 16
extremitate iniţială a unui arc
16

Forma normală a lui N.
Chomsky 92 forma normală a
lui S.
Greibach 97
formulă deschisă 59
formulă inferabilă 60 formulă
valabilă 62 formulă universal
valabilă 62 funcţie calculabilă
de o maşină Turing 165
funcţie caracteristică 14
funcţie de ieşire 73 funcţie de
proiecţie 44, 51 funcţie de
succesiune 44, 51 funcţie de
trecere 73 funcţie F-parţial
recursivă 49 funcţie nulă 44,
51 funcţie numerică parţială
44 funcţie parţială 44 funcţie
parţial calculabilă de o
maşină Turing 165 funcţie
parţial recursivă funcţie
primară 61 funcţie primară pe
alfabetul I funcţie primitiv-
recursivă 45 funcţie standard
108 funcţie totală calculabilă
de o maşină Turing 165
funcţie total definită 44
funcţie vectorială standard
109

Gradul unei maşini Turing
172 graf conex 17 graf
marcat 18 graf orientat 16
graf parţial 16 graf tare conex
17 gramatică 65 gramatică
„context free“ 68 gramatică
„context sensitive" 68
gramatică cu marcaj final #
133 gramatică de ordin n 134
gramatică de tip O 67
gramatică de tip 1 67

gramatică de tip 1 cu marcaj final # 133 gramatică de tip 2 67 gramatică de tip 2 ambiguă 91 gramatică de tip 2 neambiguă. 91 gramatică de tip 2 cu proprietatea de autoimersiune 103 gramatică de tip 3 68
gramatică expansivă 131
gramatică liberă de context 68 gramatică liniar-mărginită 135 gramatică sensibilă la context 68 gramatici echivalente 66

Hiperarbore 19 hiperarbore de deducţie 137 hipergraf 18

Infimul unei mulţimi parţial ordonate 13 injecţie 11
intersecţie de mulţimi 11
înălţimea unui vîrf 17
închiderea prin produs a unui limbaj 29
închiderea reflexivă şi tranzitivă a unei relaţii 16

Juxtapunere 28

Latice 22
latice complementată 26
latice completă 22 latice distributivă 26 latice mărginită 25 limbaj acceptat de un *ALM* prin o mulţime de stări finale 142 limbaj acceptat de o *MT* prin o mulţime de stări finale 152 limbaj acceptat de o *MTBI* prin o mulţime de stări finale 155 limbaj acceptat de o *MTMMB* prin o mulţime de stări finale 158 limbaj acceptat de o *M TND* prin o mulţime de stări finale 161 limbaj context free“ 68 limbaj „context 68 sensitive"
limbaj descris de un term 108
limbaj de tip ALGOL 110
limbaj de tip Dvck 56 limbaj de Lip 2 cu proprietatea de autoimersiune 103
limbaj generat de o gramatică 66
limbaj inerent ambiguu 91
limbaj neambiguu 91
limbaj n-parantetic 66
limbaj pe alfabetul *I* 29
limbaj recunoscut de un APD 115
limbaj regulat 76
limbaj reprezentat într-un automat iniţial prin o mulţime de stări 75
limbaj reprezentat într-un automat iniţial prin un semnal de ieşire 75
limbaj reprezentat într-un *APD* prin o mulţime de stări finale 116 limbaj reprezentat într-un *AL* 123
limbaj reprezentat într-un semiautomat nedeterminist 76 limbajul formulelor calculului predicatelor 58

limbajul formulelor calculului propoziţional 54
literă 28
lungimea unui calcul 114
lungimea unui cuvînt 28
lungimea unui drum 17
lungimea unei demonstraţii 56

Maşină Turing 150
maşină Turing bilateral infinită 154
maşină Turing cu n benzi 157
maşină Turing nedeterministă 159 maşină Turing nestaţionară 151 memorie de tip pushdown 113 metalimbaj 107
metrică 29
metrică indusă pe o normă 30
modus ponens 58
monoid 27
monoid semilaticial 28
monom ciclic 39 monom Zc-ciclic 39 monom strict ciclic 39 mulţimea arcelor 16
mulţimea funcţiilor F-elementare 44
mulţimea funcţiilor F-recursive 45
mulţimea simbolilor de marcare 18
mulţimea vidă 10 mulţime 10
mulţime cit 13 mulţime de stări 73 mulţime finită 20
mulţime infinit numărabilă 20
mulţime numărabilă 20
mulţime parţial ordonată 12
mulţime parţial recursivă 49
mulţime primitiv recursivă 49
mulţime recursivă 52 mulţime recursiv enumerabilă 52
mulţimi egale 10

Normă 30
norma unui vcctoi' 30 număr cardinal 20

Operator ataşat unei stări 75
operator de i-minimizare 52
operator de z-recursie 51
operator de închidere 15
operator de minimizare 48
operator de recursie primitivă 44 operator de superpoziţie 44 operator indus de un limbaj 34 operator nesingular 34 operaţie binară 12
operaţie de închidere prin prefixe 81 operaţie de oglindire 81 operaţie de substituţie generată de o mulţime de limbaje 82
operaţie de substituţie nesingu- lară generată de o mulţime de limbaje . 138
operaţie n-ară 11 operator contractant 32

Parte 10
partiţie 13
polinom liniar 38 polinom pe 38
principiul dualităţii pentru latici 22 produs cartezian de mulţimi 11 produs de limbaje 29 produs de relaţii 12

programul unei maşini Turing 164 proiecţia *j-a* produsului cartezian 11

Rădăcina unui arbore 17 rădăcina unui hiperarbore 19 rangul unei relaţii 13 realizare a calculului propoziţiilor 55 regulă de generare 65 relaţie antisimetrică 12 relaţie binară 11 relaţie de echivalenţă 12 relaţie de ordine 12 relaţie de preordine 12 relaţie diagonală 12 relaţie inversă 12 relaţie n-ară 11 relaţie reflexivă 12 relaţie simetrică 12 relaţie tranzitivă 12 relaţiile lui De Morgan 26 reprezentant al unei clase de echivalenţă 13 reuniune de mulţimi 11

Semiautomat ataşat unui automat 73 semigrup 27 semigrup cu unitate 27 semigrup liber 28 semigrup semilaticial 28, 39 semilatice 27 semn de punctuaţie 54 simbol 28 simbol de bază al calculului propoziţiilor 54 simbol de start 114 simbol metalingvistic 107 sistem de închidere 14 situaţie de oprire pentru o *MT* 151 soluţie minimală 37 sp - algebră 124 sp - algebră generată de o familie de limbaje 124 sp - algebră secvenţială 124 spaţiu liniar normat 30 spaţiu liniar real 29 spaţiu metric 29 spaţiu metric complet 31 stare globală a unui automat pushdown 114 stare terminală a unei mulţimi subarbore de deducţie 87 subgraf 16 subgraf parţial 16 submulţime 10 supremumul unei mulţimi parţial ordonată 13 surjecţie 11 Şir Cauchv 31 şir convergent 31

Tautologie 55 teorema de punct fix a lui Banach 31 teorema de punct fix a lui Knăster-Tarski 25 teorema de completitudine a calculului propoziţional 58 term 108 teza lui A. Church 49

Variabilă a unui term 108 vîrf al unui graf 16 vîrf al unui hipergraf 18 vîrf conectat al unui hipergraf 19 vîrf interior al unui hipergraf 19 vîrf terminal al unui

hiperarborel9

TABLA DE MATERII

Prefaţă 5

I.istă de notaţii principale..... 6

Partea I Elemente de algebră

§1.0 Introducere..... 10
§ 1.1. Noţiuni de teoria mulţimilor..... 10
§ 1.2. Latici 22
§ 1.3. Semigrupuri..... 27
§ 1.4. Spaţii metrice şi spaţii liniare..... 29
§ 1.5. Spaţiul metric al limbajelor..... 32
§ 1.6. Ecuaţii pe mulţimea limbajelor 38
§ 1.7. Funcţii recursive..... 44
§ 1.8. Elemente de logică matematică..... 54

4

Partea a Il-a Limbaje formale

§ II.O. Introducere..... 65
§ II.l. Gramatici..... 65
§ II.2. Ierarhia gramaticilor 67
§ II.3. Automate finite şi limbaje regulate.....•" 72
§ 11.4. Limbaje de tip 3..... 76
§ II.5. Limbaje de tip 2 (limbaje litere de.....context) 85
§ IÎ.6. Caracterizarea limbajelor de tip 3 In clasa limbajelor de tip 2..... 102
§ II.7. Limbaje de tip ALGOL. 106
§ II.8. Automate pushdown (automate cu.....pile de memorie) 112
§ II.9. Automate de tip Leticevski..... 122
§ 11.10. Limbaje de tip 1 (limbaje sensibile Ia context)..... 131
§ 11.11. Automate liniar mărginite 141
§ 11.12. Maşini Turing — dispozitive de recunoaştere.....a limbajelor 150
§ 11.13. Maşini Turing — dispozitive de calcul..... 164
§ 11.14. Mulţimi recursiv enumerabile şi limbaje..... 170